

GENERALIZED CRAIG INTERPOLATION FOR STOCHASTIC BOOLEAN SATISFIABILITY PROBLEMS WITH APPLICATIONS TO PROBABILISTIC STATE REACHABILITY AND REGION STABILITY

TINO TEIGE AND MARTIN FRÄNZLE

Carl von Ossietzky University of Oldenburg, Department of Computing Science, Research Group
Hybrid Systems, D-26111 Oldenburg, Germany
e-mail address: {tino.teige, fraenzle}@informatik.uni-oldenburg.de

ABSTRACT. The stochastic Boolean satisfiability (SSAT) problem has been introduced by Papadimitriou in 1985 when adding a probabilistic model of uncertainty to propositional satisfiability through randomized quantification. SSAT has many applications, among them probabilistic bounded model checking (PBMC) of symbolically represented Markov decision processes. This article identifies a notion of *Craig interpolant* for the SSAT framework and develops an algorithm for computing such interpolants based on a resolution calculus for SSAT.

As a potential application area of this novel concept of Craig interpolation, we address the symbolic analysis of probabilistic systems. We first investigate the use of interpolation in probabilistic state reachability analysis, turning the falsification procedure employing PBMC into a verification technique for probabilistic safety properties. We furthermore propose an interpolation-based approach to probabilistic region stability, being able to verify that the probability of stabilizing within some region is sufficiently large.

INTRODUCTION

Papadimitriou [Pap85] has proposed the idea of modeling uncertainty within propositional satisfiability (SAT) by adding *randomized* quantification to the problem description. The resultant *stochastic Boolean satisfiability* (SSAT) problems consist of a quantifier prefix followed by a propositional formula. The quantifier prefix is an alternating sequence of existentially quantified variables and variables bound by randomized quantifiers. The meaning of a randomized variable x is that x takes value **true** with a certain probability p and value **false** with the complementary probability $1 - p$. Due to the presence of such probabilistic assignments, the semantics of an SSAT formula Φ no longer is qualitative in the sense that

1998 ACM Subject Classification: D.2.4, F.3.1, F.4.1.

Key words and phrases: stochastic Boolean satisfiability, Craig interpolation, probabilistic state reachability, probabilistic region stability.

This work has been supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center “Automatic Verification and Analysis of Complex Systems” (SFB/TR 14 AVACS, www.avacs.org) as well as by the European Union Seventh Framework Programme FP7/2007-2013 under the MoVeS Project (grant agreement No. 257005, <http://www.movesproject.eu>).

Φ is satisfiable or unsatisfiable, but rather *quantitative* in the sense that we are interested in the *maximum probability of satisfaction* of Φ . Intuitively, a solution of Φ is a strategy for assigning the existential variables, i.e. a tree of assignments to the existential variables depending on the probabilistically determined values of preceding randomized variables, such that the assignments maximize the probability of satisfying the propositional formula.

In recent years, the SSAT framework has attracted interest within the Artificial Intelligence community, as many problems from that area involving uncertainty have concise descriptions as SSAT problems, in particular probabilistic planning problems [LMP01, ML98, ML03]. Inspired by that work, other communities have started to exploit SSAT and closely related formalisms within their domains. The Constraint Programming community is working on *stochastic constraint satisfaction* problems [Wal02, BS06] to address, among others, multi-objective decision making under uncertainty [BS07]. Recently, a technique for the symbolic analysis of probabilistic hybrid systems based on stochastic satisfiability has been suggested by the authors [FHT08, TF09, FTE10, TEF11]. To this end, SSAT has been extended by embedded theory reasoning over arithmetic theories, as known from *satisfiability modulo theories* (SMT) [BSST09], which yields the notion of *stochastic satisfiability modulo theories* (SSMT). By the expressive power of SSMT, bounded probabilistic reachability problems of uncertain hybrid systems can be phrased symbolically as SSMT formulae yielding the same probability of satisfaction [FHT08, TF09, FTE10, TEF11]. As this bounded model checking approach yields valid lower bounds lb of the probability of reaching undesirable system states along unbounded runs, it is able to *falsify* probabilistic safety requirements of shape “a system error occurs with probability at most 0.1%”, namely if a lower bound $lb > 0.1\%$ is computed.

Though the general SSAT problem and even its restriction to 2CNF, i.e. to formulae in conjunctive normal form containing clauses with two literals only, are PSPACE-complete [TF10], the plethora of real-world applications calls for practically efficient algorithms. The first SSAT algorithm, suggested by Littman [Lit99], extends the Davis-Putnam-Logemann-Loveland (DPLL) procedure [DP60, DLL62] for SAT with appropriate quantifier handling and algorithmic optimizations like *thresholding*. Majercik further improved the DPLL-based SSAT algorithm by *non-chronological backtracking* [Maj04]. The SSMT algorithm from [FHT08, TF08, TF09, TEF11] being implemented in the SSMT tool SiSAT builds on the DPLL-based SSAT procedures plus conflict-driven clause learning, but also integrates an underlying theory solver addressing non-linear arithmetics, and was successfully applied to realistic case studies featuring hybrid discrete-continuous state spaces [TF09, FTE10, TEF11]. Unlike these explicit tree-traversal approaches and motivated by work on *resolution* for propositional and first-order formulae [Rob65] and for quantified Boolean formulae (QBF) [BKF95], the authors have recently developed an alternative SSAT procedure based on resolution [TF10].

In this article, we investigate the concept of Craig interpolation for SSAT. Given two formulae A and B for which $A \Rightarrow B$ is true, a *Craig interpolant* [Cra57] \mathcal{I} is a formula over variables common to A and B that “lies in between” A and B in the sense that $A \Rightarrow \mathcal{I}$ and $\mathcal{I} \Rightarrow B$. In the automatic hardware and software verification communities, Craig interpolation has found widespread use in model checking algorithms, both as a means of extracting reasons for non-concretizability of a counterexample obtained on an abstraction as well as for obtaining a symbolic description of reachable state sets. In McMillan’s approach [McM03, McM05], interpolants are used to symbolically describe an overapproximation of the step-bounded reachable state set. If the sequence of interpolants

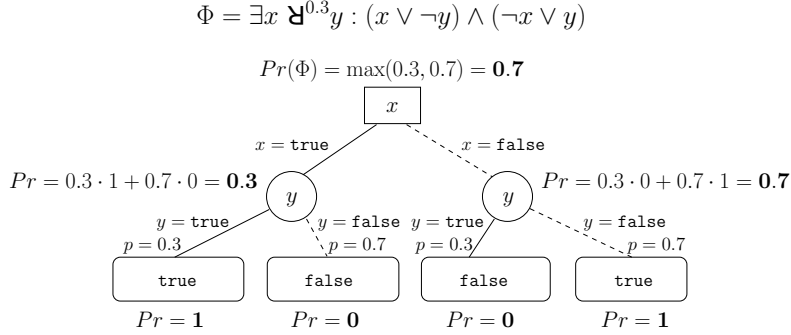


Figure 1: Semantics of an SSAT formula depicted as a tree.

thus obtained stabilizes eventually, i.e. no additional state is found to be reachable, then the corresponding state-set predicate R has all reachable system states as its models. The safety property that states satisfying B , where B is a predicate, are never reachable is then verified by checking $R \wedge B$ for unsatisfiability.

Given McMillan’s verification approach to reachability analysis of non-probabilistic systems based on Craig interpolation for SAT, it is natural to ask whether a corresponding probabilistic counterpart can be developed, i.e. a *verification approach to probabilistic reachability analysis of probabilistic systems based on Craig interpolation for stochastic SAT*. Such an approach would complement the aforementioned falsification procedure for probabilistic systems based on SSAT/SSMT. In this article, we suggest a solution to the issue above.

In addition to probabilistic state reachability, we address the problem of *probabilistic region stability*. The latter problem is motivated by the notion of region stability for non-probabilistic hybrid systems [PW07a, PW07b], where a system is called stable with respect to some region R iff all system runs eventually reach R and finally stay in R forever. In this article, we suggest an adaptation of region stability to the probabilistic case along with a symbolic, interpolation-based procedure for the verification of probabilistic stability properties like “the probability that the system stabilizes within region R is at least 99.9%”.

Structure of the article. After a formal introduction to SSAT in Section 1, Section 2 is devoted to a generalization of the notion of Craig interpolants suitable for SSAT. Thereafter, Section 3 elaborates on an algorithm for computing such generalized Craig interpolants, which relies on a resolution calculus for SSAT. The application of generalized Craig interpolation to the symbolic analysis of probabilistic systems, namely to probabilistic state reachability as well as to probabilistic region stability, is then addressed in Section 4, where applicability of these novel techniques is illustrated on small examples. Section 5 finally concludes the article.

1. STOCHASTIC BOOLEAN SATISFIABILITY

A *stochastic Boolean satisfiability* (SSAT) formula is of the form $\Phi = \mathcal{Q} : \varphi$ with a prefix $\mathcal{Q} = Q_1 x_1 \dots Q_n x_n$ of quantified propositional variables x_i , where Q_i is either an existential quantifier \exists or a randomized quantifier \mathfrak{P}^{p_i} with a rational constant $0 < p_i < 1$, and a propositional formula φ such that $Var(\varphi) \subseteq \{x_1, \dots, x_n\}$, where $Var(\varphi)$ denotes the set of all (necessarily free) variables occurring in φ . Note that SSAT formula Φ thus has no free variables. Without loss of generality, we assume that φ is in *conjunctive normal*

form (CNF), i.e. a conjunction of disjunctions of propositional literals. A *literal* ℓ is a propositional variable, i.e. $\ell = x_i$, or its negation, i.e. $\ell = \neg x_i$. A *clause* is a disjunction of literals. Throughout the article and without loss of generality, we require that a clause does not contain the same literal more than once as $\ell \vee \ell \equiv \ell$. Consequently, we may also identify a clause with its set of literals. The semantics of Φ , as illustrated in Figure 1, is defined by the *maximum probability of satisfaction* $Pr(\Phi)$ as follows.

$$\begin{aligned} Pr(\varepsilon : \varphi) &= \begin{cases} 0 & \text{if } \varphi \text{ is logically equivalent to } \mathbf{false} \\ 1 & \text{if } \varphi \text{ is logically equivalent to } \mathbf{true} \end{cases} \\ Pr(\exists x \ Q : \varphi) &= \max(Pr(Q : \varphi[\mathbf{true}/x]), Pr(Q : \varphi[\mathbf{false}/x])) \\ Pr(\forall^p x \ Q : \varphi) &= p \cdot Pr(Q : \varphi[\mathbf{true}/x]) + (1 - p) \cdot Pr(Q : \varphi[\mathbf{false}/x]) \end{aligned}$$

Note that the semantics is well-defined as Φ has no free variables such that all variables have been substituted by the constants **true** and **false** when reaching the quantifier-free base case.

2. GENERALIZED CRAIG INTERPOLANTS

Craig interpolation [Cra57] is a well-studied notion in formal logics which has several applications in Computer Science, among them model checking [McM03, McM05]. Given two formulae φ and ψ such that $\varphi \Rightarrow \psi$ is valid, a *Craig interpolant* for (φ, ψ) is a formula \mathcal{I} which refers only to common variables of φ and ψ , and \mathcal{I} is “intermediate” in the sense that $\varphi \Rightarrow \mathcal{I}$ and $\mathcal{I} \Rightarrow \psi$. Such interpolants do trivially exist in all logics permitting quantifier elimination, for instance, in propositional logic. The observation that $\varphi \Rightarrow \psi$ holds iff $\varphi \wedge \neg\psi$ is unsatisfiable gives rise to an equivalent definition which we refer to in the rest of the article:¹ given an unsatisfiable formula $\varphi \wedge \neg\psi$, a formula \mathcal{I} is a Craig interpolant for (φ, ψ) iff both $\varphi \wedge \neg\mathcal{I}$ and $\mathcal{I} \wedge \neg\psi$ are unsatisfiable and \mathcal{I} mentions only common variables.

In this section, we investigate the issue of Craig interpolation for stochastic SAT. We propose a generalization of Craig interpolants suitable for SSAT and show the general existence of such interpolants. In Section 3, we then devote our attention to an automatic method for computing generalized Craig interpolants based on a resolution calculus for SSAT.

When approaching a reasonable definition of interpolants for SSAT, the semantics of the non-classical quantifier prefix poses problems: Let $\Phi = Q : (A \wedge B)$ be an SSAT formula. Each variable in $A \wedge B$ is bound by Q , which provides the probabilistic interpretation of the variables that is lacking without the quantifier prefix. This issue can be addressed by considering the quantifier prefix Q as the global setting that serves to interpret the quantifier-free part, and consequently interpreting the interpolant also within the scope of Q , thus reasoning about $Q : (A \wedge \neg\mathcal{I})$ and $Q : (\mathcal{I} \wedge B)$. A more fundamental problem is that a classical Craig interpolant for Φ only exists if $Pr(\Phi) = 0$, since $A \wedge B$ has to be unsatisfiable by definition of a Craig interpolant which applies iff $Pr(Q : (A \wedge B)) = 0$. The precondition that $Pr(Q : (A \wedge B)) = 0$ would be far too restrictive for application of interpolation, as the notion of unsatisfiability of $A \wedge B$ is naturally generalized to satisfiability with insufficient probability, i.e. $Pr(Q : (A \wedge B))$ being “sufficiently small”, in the stochastic setting. Such

¹This is of technical nature as SSAT formulae are interpreted by the maximum probability of satisfaction. As the *maximum* probability that an implication $\varphi \Rightarrow \psi$ holds is inappropriate for our purpose, we reason about the maximum satisfaction probability p of the negated implication, i.e. of $\varphi \wedge \neg\psi$, instead. The latter relates to the *minimum* probability $1 - p$ that $\varphi \Rightarrow \psi$ holds, which is the desired notion.

relaxed requirements actually appear in practice, for instance, in probabilistic verification where safety properties like “a fatal system error is never reachable” are frequently replaced by probabilistic ones like “a fatal system error is reachable only with (sufficiently small) probability of at most 0.1%”. Motivated by above facts, interpolants for SSAT should also exist when $A \wedge B$ is satisfiable with reasonably low probability.

The resulting notion of interpolation, which is to be made precise in Definition 2.1, matches the following intuition. In classical Craig interpolation, when performed in logics permitting quantifier elimination, the Craig interpolants of $(A, \neg B)$ form a lattice with implication as its ordering, $A^\exists = \exists a_1, \dots, a_\alpha : A$ as its bottom element and $\overline{B}^\forall = \neg \exists b_1, \dots, b_\beta : B$ as its top element, where the a_i and b_i are the local variables of A and of B , respectively. In the generalized setting required for SSAT², $A \Rightarrow \neg B$ and thus $A^\exists \Rightarrow \overline{B}^\forall$ may no longer hold such that the above lattice can collapse to the empty set. To preserve the overall structure, it is however natural to use the lattice of propositional formulae “in between” $A^\exists \wedge \overline{B}^\forall$ as bottom element and $A^\exists \vee \overline{B}^\forall$ as top element instead. This lattice is non-empty and coincides with the classical one whenever $A \wedge B$ is unsatisfiable.

Definition 2.1 (Generalized Craig interpolant). Let A, B be propositional formulae and $V_A := \text{Var}(A) \setminus \text{Var}(B) = \{a_1, \dots, a_\alpha\}$, $V_B := \text{Var}(B) \setminus \text{Var}(A) = \{b_1, \dots, b_\beta\}$, $V_{A,B} := \text{Var}(A) \cap \text{Var}(B)$, $A^\exists = \exists a_1, \dots, a_\alpha : A$, and $\overline{B}^\forall = \neg \exists b_1, \dots, b_\beta : B$. A propositional formula \mathcal{I} is called *generalized Craig interpolant for (A, B)* iff $\text{Var}(\mathcal{I}) \subseteq V_{A,B}$, $(A^\exists \wedge \overline{B}^\forall) \Rightarrow \mathcal{I}$, and $\mathcal{I} \Rightarrow (A^\exists \vee \overline{B}^\forall)$.

Given any two propositional formulae A and B , the four quantifier-free propositional formulae equivalent to $A^\exists \wedge \overline{B}^\forall$, to A^\exists , to \overline{B}^\forall , and to $A^\exists \vee \overline{B}^\forall$, are generalized Craig interpolants for (A, B) . These generalized interpolants always exist since propositional logic has quantifier elimination.

While Definition 2.1 motivates the generalized notion of Craig interpolant from a model-theoretic perspective, we state an equivalent definition of generalized Craig interpolants in Lemma 2.2 that substantiates the intuition of generalized interpolants and allows for an illustration of their geometric shape. Given two formulae A and B , the idea of generalized Craig interpolant is depicted in Figure 2. The set of solutions of A is defined by the rectangle on the $V_A, V_{A,B}$ -plane with a cylindrical extension in V_B -direction as A does not contain variables in V_B . Similarly, the solution set of B is given by the triangle on the $V_B, V_{A,B}$ -plane and its cylinder in V_A -direction. The solution set of $A \wedge B$ is then determined by the intersection of both cylinders. Since $A \wedge B \wedge \neg(A \wedge B)$ is unsatisfiable, the sets $A \wedge \neg(A \wedge B)$ and $B \wedge \neg(A \wedge B)$ are disjoint. This gives us the possibility to talk about interpolants wrt. these sets. However, a formula \mathcal{I} over only common variables in $V_{A,B}$ may not exist when demanding $A \wedge \neg(A \wedge B) \wedge \neg \mathcal{I}$ and $\mathcal{I} \wedge B \wedge \neg(A \wedge B)$ to be unsatisfiable. This is indicated by Figure 2 and proven by the simple example $A = (a)$, $B = (b)$. As $V_{A,B} = \emptyset$, \mathcal{I} is either **true** or **false**. In first case, **true** $\wedge (b) \wedge \neg(a \wedge b)$ is satisfiable, while $(a) \wedge \neg(a \wedge b) \wedge \neg \mathbf{false}$ is in second case. If we however project the solution set of $A \wedge B$ onto the $V_{A,B}$ -axis and subtract the resulting hyperplane $\mathcal{S}_{A,B}$ from A and B then such a formula \mathcal{I} over $V_{A,B}$ -variables exists. The next lemma formalizes such generalized interpolants \mathcal{I} and shows their equivalence to the ones from Definition 2.1.

²Though the concept seems to be more general, this article addresses SSAT only.

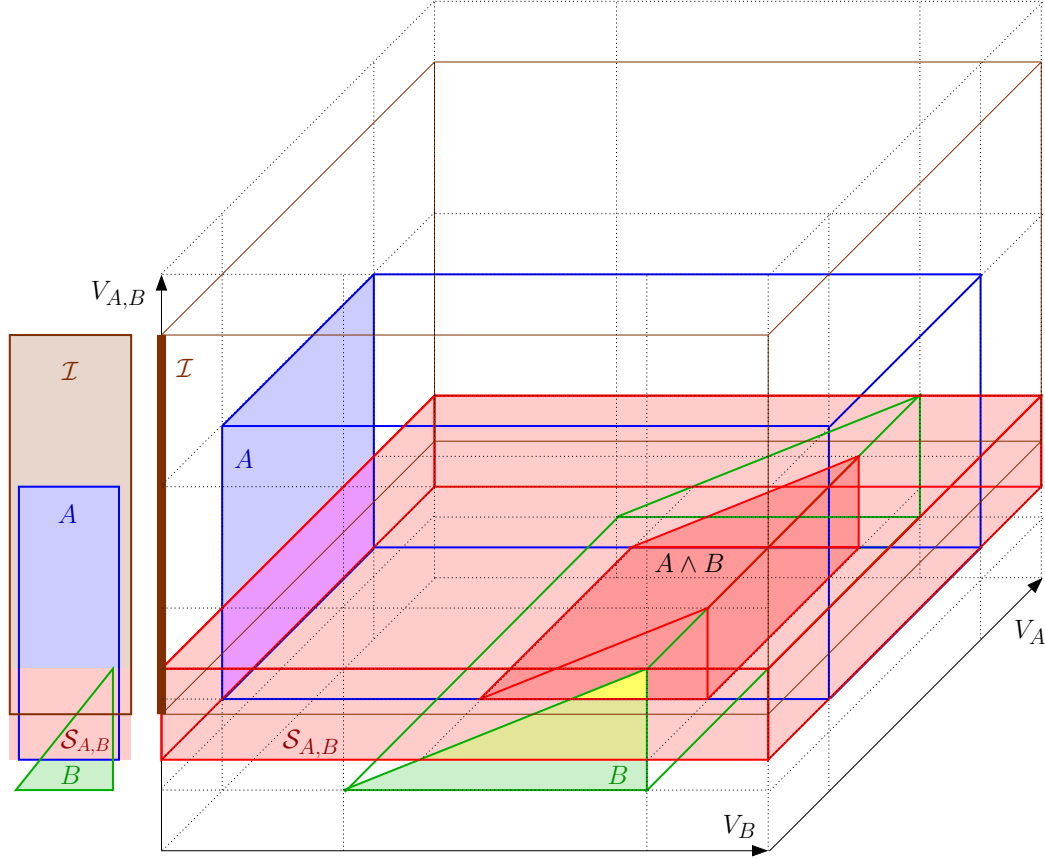


Figure 2: Geometric interpretation of a generalized Craig interpolant \mathcal{I} . V_A -, V_B -, and $V_{A,B}$ -axes denote assignments of variables occurring only in A , only in B , and in both A and B , respectively.

Lemma 2.2 (Generalized Craig interpolant for SSAT). *Let $\Phi = \mathcal{Q} : (A \wedge B)$ be some SSAT formula, V_A , V_B , $V_{A,B}$ be defined as in Definition 2.1, and $\mathcal{S}_{A,B}$ be a propositional formula with $\text{Var}(\mathcal{S}_{A,B}) \subseteq V_{A,B}$ such that $\mathcal{S}_{A,B} \equiv \exists a_1, \dots, a_\alpha, b_1, \dots, b_\beta : (A \wedge B)$. Then, a propositional formula \mathcal{I} is a generalized Craig interpolant for (A, B) iff the following properties are satisfied.*

- (1) $\text{Var}(\mathcal{I}) \subseteq V_{A,B}$
- (2) $\text{Pr}(\mathcal{Q} : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg \mathcal{I})) = 0$
- (3) $\text{Pr}(\mathcal{Q} : (\mathcal{I} \wedge B \wedge \neg \mathcal{S}_{A,B})) = 0$

Proof. As $\text{Var}(\mathcal{I}) \subseteq V_{A,B}$ holds for generalized Craig interpolants \mathcal{I} , it remains to show that $(A^\exists \wedge \overline{B}^\forall) \Rightarrow \mathcal{I}$ and $\mathcal{I} \Rightarrow (A^\exists \vee \overline{B}^\forall)$ iff $\text{Pr}(\mathcal{Q} : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg \mathcal{I})) = 0$ and $\text{Pr}(\mathcal{Q} : (\mathcal{I} \wedge B \wedge \neg \mathcal{S}_{A,B})) = 0$. Observe that $\models (A^\exists \wedge \overline{B}^\forall) \Rightarrow \mathcal{I}$ iff $\models \forall a_1, \dots, a_\alpha : (A \wedge \overline{B}^\forall) \Rightarrow \mathcal{I}$ iff $\models (A \wedge \overline{B}^\forall) \Rightarrow \mathcal{I}$ iff $\models (A \wedge (\neg A^\exists \vee \overline{B}^\forall)) \Rightarrow \mathcal{I}$ iff $\models (A \wedge \neg \mathcal{S}_{A,B}) \Rightarrow \mathcal{I}$ iff $A \wedge \neg \mathcal{S}_{A,B} \wedge \neg \mathcal{I}$ is unsatisfiable iff $\text{Pr}(\mathcal{Q} : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg \mathcal{I})) = 0$. Analogously, $\models \mathcal{I} \Rightarrow (A^\exists \vee \overline{B}^\forall)$ iff $\models \forall b_1, \dots, b_\beta : \mathcal{I} \Rightarrow (A^\exists \vee \neg B)$ iff $\models \mathcal{I} \Rightarrow (A^\exists \vee \neg B)$ iff $\models \mathcal{I} \Rightarrow ((A^\exists \wedge \neg \overline{B}^\forall) \vee \neg B)$ iff $\models \mathcal{I} \Rightarrow (\mathcal{S}_{A,B} \vee \neg B)$ iff $\mathcal{I} \wedge \neg \mathcal{S}_{A,B} \wedge B$ is unsatisfiable iff $\text{Pr}(\mathcal{Q} : (\mathcal{I} \wedge B \wedge \neg \mathcal{S}_{A,B})) = 0$. \square

We remark that the concept of generalized Craig interpolants is a generalization of Craig interpolants in the sense that whenever $A \wedge B$ is unsatisfiable, i.e. when $Pr(\mathcal{Q} : (A \wedge B)) = 0$, then each generalized Craig interpolant \mathcal{I} for (A, B) actually is a Craig interpolant for A and B since $\mathcal{S}_{A,B} \equiv \text{false}$.

3. COMPUTATION OF GENERALIZED CRAIG INTERPOLANTS

In this section, we proceed to the efficient computation of generalized Craig interpolants. The remark following Definition 2.1 shows that generalized interpolants can in principle be computed by *explicit* quantifier elimination methods, like Shannon's expansion or binary decision diagrams (BDDs). We aim at a more efficient method based on SSAT resolution [TF10] akin to resolution-based Craig interpolation for propositional SAT by Pudlák [Pud97]. The latter approach has been integrated into DPLL-based SAT solvers featuring conflict analysis and successfully applied to symbolic model checking [McM03, McM05]. To this end, we first recall the sound and complete resolution calculus for SSAT from [TF10] in Section 3.1. Thereafter, SSAT resolution is enhanced in order to compute generalized Craig interpolants in Section 3.2.

3.1. Resolution for SSAT. As basis of the SSAT interpolation procedure introduced in Section 3.2, we recall the sound and complete resolution calculus for SSAT from [TF10], subsequently called *S-resolution*. In contrast to SSAT algorithms implementing a DPLL-based backtracking procedure, thereby explicitly traversing the tree given by the quantifier prefix and recursively computing the individual satisfaction probabilities for each subtree by the scheme illustrated in Figure 1, S-resolution follows the idea of *resolution* for propositional and first-order formulae [Rob65] and for QBF formulae [BKF95] by deriving new clauses c^p annotated with probabilities $0 \leq p \leq 1$. S-resolution differs from non-stochastic resolution, as such derived clauses c^p need not be implications of the given formula, but are just entailed with some probability. Informally speaking, the derivation of a clause c^p means that under SSAT formula $\mathcal{Q} : \varphi$, the clause c is violated with a maximum probability at most p , i.e. the satisfaction probability of $\mathcal{Q} : (\varphi \wedge \neg c)$ is at most p . More intuitively, the minimum probability that clause c is implied by φ is at least $1 - p$.³ Once an annotated empty clause \emptyset^p is derived, it follows that the probability of the given SSAT formula is at most p , i.e. $Pr(\mathcal{Q} : (\varphi \wedge \neg \text{false})) = Pr(\mathcal{Q} : \varphi) \leq p$.

In what follows, let $\mathcal{Q} : \varphi$ be an SSAT formula with φ in CNF. Without loss of generality, φ contains only non-tautological clauses⁴, i.e. $\forall c \in \varphi : \not\models c$. Let $\mathcal{Q} = Q_1 x_1 \dots Q_n x_n$ be the quantifier prefix and φ be some propositional formula with $Var(\varphi) \subseteq \{x_1, \dots, x_n\}$. The quantifier prefix $\mathcal{Q}(\varphi)$ is defined to be shortest prefix of \mathcal{Q} that contains all variables from φ , i.e. $\mathcal{Q}(\varphi) = Q_1 x_1 \dots Q_i x_i$ where $x_i \in Var(\varphi)$ and for each $j > i : x_j \notin Var(\varphi)$. Let further be $Var(\varphi) \downarrow_k := \{x_1, \dots, x_k\}$ for each integer $0 \leq k \leq n$. For a non-tautological clause c , i.e. if $\not\models c$, we define the unique assignment $\mathbb{f}\mathbb{f}_c$ that falsifies c as the mapping

$$\mathbb{f}\mathbb{f}_c : Var(c) \rightarrow \mathbb{B} \text{ such that } \forall x \in Var(c) : \mathbb{f}\mathbb{f}_c(x) = \begin{cases} \text{true} & ; \neg x \in c, \\ \text{false} & ; x \in c. \end{cases}$$

Consequently, c evaluates to **false** under assignment $\mathbb{f}\mathbb{f}_c$.

³We remark that $Pr(\mathcal{Q} : \psi) = 1 - Pr(\mathcal{Q}' : \neg\psi)$, where \mathcal{Q}' arises from \mathcal{Q} by replacing existential quantifiers by universal ones, where universal quantifiers call for *minimizing* the satisfaction probability.

⁴Tautological clauses c , i.e. $\models c$, are redundant, i.e. $Pr(\mathcal{Q} : (\varphi \wedge c)) = Pr(\mathcal{Q} : \varphi)$.

Starting with clauses in φ , *S-resolution* is given by the consecutive application of rules **R.1** to **R.3** to derive new clauses c^p with $0 \leq p \leq 1$. Rule **R.1** derives a clause c^0 from an original clause c in φ . Referring to the definition of $Pr(Q : \varphi)$ in Section 1, **R.1** corresponds to the quantifier-free base case where φ is equivalent to **false** under any assignment that falsifies c .

$$\frac{c \in \varphi}{c^0} \quad (\text{R.1})$$

Similarly, **R.2** reflects the quantifier-free base case in which φ is equivalent to **true** under any assignment τ' that is conform to the partial assignment τ since $\models \varphi[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]$. The constructed clause c^1 then encodes the opposite of this satisfying (partial) assignment τ . We remark that finding such a τ in the premise of **R.2** is NP-hard (equivalent to finding a solution of a propositional formula in CNF). This strong condition on τ is not essential for soundness and completeness and could be removed⁵ but, as mentioned above, facilitates a less technical presentation of generalized interpolation in Section 3.2. Another argument justifying the strong premise of **R.2** is a potential integration of S-resolution into DPLL-based SSAT solvers since whenever a satisfying (partial) assignment τ of φ is found by an SSAT solver then τ meets the requirements of **R.2**.

$$\frac{\begin{array}{l} c \subseteq \{x, \neg x \mid x \in \text{Var}(\varphi)\}, \not\models c, Q(c) = Q_1 x_1 \dots Q_i x_i, \\ \text{for each } \tau : \text{Var}(\varphi) \downarrow_i \rightarrow \mathbb{B} \text{ with } \forall x \in \text{Var}(c) : \tau(x) = \text{ff}_c(x) : \\ \quad \models \varphi[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i] \end{array}}{c^1} \quad (\text{R.2})$$

Rule **R.3** finally constitutes the actual resolution rule as known from the non-stochastic case. Depending on whether an existential or a randomized variable is resolved upon, the probability value of the resolvent clause is computed according to the semantics $Pr(Q : \varphi)$ defined in Section 1.

$$\frac{\begin{array}{l} (c_1 \vee \neg x)^{p_1}, (c_2 \vee x)^{p_2}, Qx \in Q, Qx \notin Q(c_1 \vee c_2), \not\models (c_1 \vee c_2), \\ p = \begin{cases} \max(p_1, p_2) & ; Q = \exists \\ p_x \cdot p_1 + (1 - p_x) \cdot p_2 & ; Q = \mathfrak{A}^{p_x} \end{cases} \end{array}}{(c_1 \vee c_2)^p} \quad (\text{R.3})$$

The derivation of a clause c^p by **R.1** from c , by **R.2**, and by **R.3** from $c_1^{p_1}, c_2^{p_2}$ is denoted by $c \vdash_{\text{R.1}} c^p$, by $\vdash_{\text{R.2}} c^p$, and by $(c_1^{p_1}, c_2^{p_2}) \vdash_{\text{R.3}} c^p$, respectively. Given rules **R.1** to **R.3**, S-resolution is sound and complete in the following sense.

Lemma 3.1. *Let clause c^p be derivable by S-resolution and let $Q(c) = Q_1 x_1 \dots Q_i x_i$. For each $\tau : \text{Var}(\varphi) \downarrow_i \rightarrow \mathbb{B}$ with $\forall x \in \text{Var}(c) : \tau(x) = \text{ff}_c(x)$ it holds that $Pr(Q_{i+1} x_{i+1} \dots Q_n x_n : \varphi[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = p$.*

Proof. We show the lemma by induction over the application of rules **R.1**, **R.2**, and **R.3**. The base case is given by rules **R.1** and **R.2**. By construction of τ , $\varphi[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]$ is unsatisfiable for **R.1** and tautological for **R.2** which immediately establishes the result for the base case. Now assume that the assumption holds for all clauses in the premises of **R.3**, i.e.

$$\begin{aligned} Pr(Q_{j+1} x_{j+1} \dots Q_n x_n : \varphi[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\text{true}/x_j]) &= p_1, \\ Pr(Q_{j+1} x_{j+1} \dots Q_n x_n : \varphi[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\text{false}/x_j]) &= p_2, \end{aligned}$$

⁵Then, Lemma 3.1 must be weakened to $Pr(Q_{i+1} x_{i+1} \dots Q_n x_n : \varphi[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) \leq p$, as for original S-resolution [TF10].

where $x_j = x$ with $j \geq i+1$. By definition of Pr , for each τ with $\tau(x) = \tau_1(x)$ if $x \in \text{Var}(c_1)$ and $\tau(x) = \tau_2(x)$ if $x \in \text{Var}(c_2)$ we then have

$$Pr(Q_j x_j \ Q_{j+1} x_{j+1} \dots Q_n x_n : \varphi[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) = p.$$

The result is obvious for $j = i+1$. For $j > i+1$, note that variables x_{i+1}, \dots, x_{j-1} do not occur in the derived clause $(c_1 \vee c_2)$. Hence, for $k = j-1$ down to $i+1$ we successively conclude that

$$\begin{aligned} Pr(Q_{k+1} x_{k+1} \dots Q_n x_n : \varphi[\tau(x_1)/x_1] \dots [\tau(x_{k-1})/x_{k-1}][\mathbf{true}/x_k]) &= p, \\ Pr(Q_{k+1} x_{k+1} \dots Q_n x_n : \varphi[\tau(x_1)/x_1] \dots [\tau(x_{k-1})/x_{k-1}][\mathbf{false}/x_k]) &= p. \end{aligned}$$

From case $k = i+1$ the lemma follows. \square

Corollary 3.2 (Soundness of S-resolution). *If the empty clause \emptyset^p is derivable by S-resolution from a given SSAT formula $\mathcal{Q} : \varphi$ then $Pr(\mathcal{Q} : \varphi) = p$.* \square

Corollary 3.2 follows directly from Lemma 3.1, namely for the special case $c^p = \emptyset^p$. Theorem 3.3 shows completeness of S-resolution.

Theorem 3.3 (Completeness of S-resolution). *If $Pr(\mathcal{Q} : \varphi) = p$ for some SSAT formula $\mathcal{Q} : \varphi$ then the empty clause \emptyset^p is derivable from $\mathcal{Q} : \varphi$ by S-resolution.*

Proof. If $\emptyset \in \varphi$, i.e. φ contains the empty clause, then $p = 0$ and the empty clause \emptyset^0 is derivable by rule **R.1**. In the remaining proof, we assume that $\emptyset \notin \varphi$. We prove the theorem by induction over the number of quantifiers in the quantifier prefix \mathcal{Q} . For the base case $\mathcal{Q} = Qx$ we distinguish three cases: 1) $\varphi = (\neg x) \wedge (x)$. Then $p = 0$, and $(\neg x)^0, (x)^0$ are derivable by **R.1**, and **R.3** finally yields \emptyset^0 . 2) $\varphi = (\neg x)$. Clauses $(\neg x)^0$ and $(x)^1$ are derivable by **R.1** and **R.2**, respectively, the latter since $\models \varphi[\mathbf{false}/x]$. If $Q = \exists$ or $Q = \forall^{p_x}$ then $p = 1$ or $p = (1 - p_x)$, and \emptyset^1 or $\emptyset^{(1-p_x)}$ can be derived by **R.3**, respectively. 3) $\varphi = (x)$. Analogously to 2), if $Q = \exists$ or $Q = \forall^{p_x}$ then $p = 1$ or $p = p_x$, and \emptyset^1 or \emptyset^{p_x} can be derived by **R.3**, respectively.

In the induction step, we show that \emptyset^p is derivable for $Pr(Qx \mathcal{Q} : \varphi) = p$. Let $p_1 = Pr(\mathcal{Q} : \varphi[\mathbf{true}/x])$ and $p_2 = Pr(\mathcal{Q} : \varphi[\mathbf{false}/x])$. Induction hypothesis assumes that \emptyset^{p_1} and \emptyset^{p_2} are derivable from $\mathcal{Q} : \varphi[\mathbf{true}/x]$ and $\mathcal{Q} : \varphi[\mathbf{false}/x]$. Applying the resolution sequence deriving \emptyset^{p_1} from $\mathcal{Q} : \varphi[\mathbf{true}/x]$ on $Qx \mathcal{Q} : \varphi$ yields either \emptyset^{p_1} or $(\neg x)^{p_1}$. Analogously, either \emptyset^{p_2} or $(x)^{p_2}$ is derivable from $Qx \mathcal{Q} : \varphi$. If \emptyset^{p_1} (respectively, \emptyset^{p_2}) was derived then $p = p_1$ (respectively, $p = p_2$) by Corollary 3.2. (Note that if both \emptyset^{p_1} and \emptyset^{p_2} are derivable then $p_1 = p_2$.) Otherwise, i.e. $(\neg x)^{p_1}$ and $(x)^{p_2}$ are derived, application of **R.3** gives \emptyset^p . \square

The above presentation of S-resolution differs slightly from [TF10] in order to avoid overhead in interpolant generation incurred when employing the original definition, like the necessity of enforcing particular resolution sequences. For readers familiar with [TF10], the particular modifications are: 1) derived clauses c^p may also carry value $p = 1$, 2) former rules **R.2** and **R.5** are joined into the new rule **R.2**, and 3) former rules **R.3** and **R.4** are collapsed into rule **R.3**. These modifications do not affect soundness and completeness of S-resolution, confer Corollary 3.2 and Theorem 3.3. The advantage of the modification is that derivable clauses c^p are forced to have a tight bound p in the sense that under each assignment which falsifies c , the satisfaction probability of the remaining subproblem *exactly* is p , confer Lemma 3.1. This fact confirms the conjecture from [TF10, page 14] about the existence of such clauses $(c \vee \ell)^p$ and allows for a generalized clause learning scheme to be integrated into DPLL-SSAT solvers: the idea is that under a partial assignment falsifying

c , one may directly propagate literal ℓ as the satisfaction probability of the other branch, for which the negation of ℓ holds, is known to be p already.

Example of S-resolution. Consider the SSAT formula $\Phi = \mathfrak{A}^{0.8}x_1 \exists x_2 \mathfrak{A}^{0.3}x_3 : ((x_1 \vee x_2) \wedge (\neg x_2) \wedge (x_2 \vee x_3))$ with $Pr(\Phi) = 0.24$. Clauses $(x_1 \vee x_2)^0$, $(\neg x_2)^0$, $(x_2 \vee x_3)^0$ are then derivable by **R.1**. As $x_1 = \mathbf{true}, x_2 = \mathbf{false}, x_3 = \mathbf{true}$ is a satisfying assignment, $\vdash_{\mathbf{R.2}} (\neg x_1 \vee x_2 \vee \neg x_3)^1$. Then, $((\neg x_1 \vee x_2 \vee \neg x_3)^1, (x_2 \vee x_3)^0) \vdash_{\mathbf{R.3}} (\neg x_1 \vee x_2)^{0.3}$, $((\neg x_2)^0, (\neg x_1 \vee x_2)^{0.3}) \vdash_{\mathbf{R.3}} (\neg x_1)^{0.3}$, $((\neg x_2)^0, (x_1 \vee x_2)^0) \vdash_{\mathbf{R.3}} (x_1)^0$, and finally $((\neg x_1)^{0.3}, (x_1)^0) \vdash_{\mathbf{R.3}} \emptyset^{0.24}$.

3.2. Interpolating resolution for SSAT. We now devote our attention to the computation of generalized Craig interpolants for SSAT by means of an enhanced version of S-resolution, which is akin to resolution-based Craig interpolation for propositional SAT by Pudlák [Pud97]. We remark that on SSAT formulae $\mathcal{Q} : (A \wedge B)$, Pudlák’s algorithm, which has unsatisfiability of $A \wedge B$ as precondition, will not work in general. When instead considering the unsatisfiable formula $A \wedge B \wedge \neg \mathcal{S}_{A,B}$ with $\neg \mathcal{S}_{A,B}$ in CNF then Pudlák’s method would be applicable and would actually produce a generalized Craig interpolant. The main drawback of this approach however is the explicit construction of $\neg \mathcal{S}_{A,B}$, calling for explicit quantifier elimination.

In the following, we propose an algorithm based on S-resolution for computing generalized Craig interpolants which operates directly on $A \wedge B$ without adding $\neg \mathcal{S}_{A,B}$, and thus does not comprise any preprocessing involving quantifier elimination. For this purpose, the rules of S-resolution are enhanced to deal with pairs (c^p, I) of annotated clauses c^p and propositional formulae I . Such formulae I are in a certain sense *intermediate* generalized interpolants, i.e. generalized interpolants for subformulae arising from instantiating some variables by partial assignments that falsify c , confer Lemma 3.4. Once a pair (\emptyset^p, I) comprising the empty clause is derived, I thus is a generalized Craig interpolant for the given SSAT formula. This augmented S-resolution, which we call *interpolating S-resolution*, is defined by rules **RI.1**, **RI.2**, and **RI.3**. The construction of intermediate interpolants I in **RI.1** and **RI.3** coincides with the classical rules by Pudlák [Pud97], while **RI.2** misses a corresponding counterpart. The rationale is that **RI.2** (or rather **R.2**) refers to satisfying valuations τ of $A \wedge B$, which do not exist in classical interpolation. As $A \wedge B$ becomes a tautology after substituting the partial assignment τ from **R.2** into it, its quantified variant $\mathcal{S}_{A,B} = \exists a_1, \dots, b_1, \dots : A \wedge B$ also becomes tautological under the same substitution $\mathcal{S}_{A,B}[\tau(x_1)/x_1, \dots, \tau(x_i)/x_i]$. Consequently, $\neg \mathcal{S}_{A,B}[\tau(x_1)/x_1, \dots, \tau(x_i)/x_i]$ is unsatisfiable, and so are $(A \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1, \dots, \tau(x_i)/x_i]$ and $(B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1, \dots, \tau(x_i)/x_i]$. This implies that the actual intermediate interpolant in **RI.2** can be chosen arbitrarily over variables in $V_{A,B}$. This freedom will allow us to control the geometric extent of generalized interpolants within the “don’t care”-region provided by the models of $\mathcal{S}_{A,B}$, confer Corollary 3.6.

$$\frac{c \vdash_{\mathbf{R.1}} c^p, I = \begin{cases} \mathbf{false} & ; c \in A \\ \mathbf{true} & ; c \in B \end{cases}}{(c^p, I)} \quad (\mathbf{RI.1})$$

$$\frac{\vdash_{\mathbf{R.2}} c^p, I \text{ is any formula over } V_{A,B}}{(c^p, I)} \quad (\mathbf{RI.2})$$

$$\begin{array}{c}
((c_1 \vee \neg x)^{p_1}, I_1), ((c_2 \vee x)^{p_2}, I_2), \\
((c_1 \vee \neg x)^{p_1}, (c_2 \vee x)^{p_2}) \vdash_{\mathbf{R.3}} (c_1 \vee c_2)^p, \\
I = \left\{ \begin{array}{ll} I_1 \vee I_2 & ; x \in V_A \\ I_1 \wedge I_2 & ; x \in V_B \\ (\neg x \vee I_1) \wedge (x \vee I_2) & ; x \in V_{A,B} \end{array} \right. \\
\hline
((c_1 \vee c_2)^p, I)
\end{array} \tag{RI.3}$$

The following lemma establishes the theoretical foundation of computing generalized Craig interpolants by interpreting the derived pairs (c^p, I) .

Lemma 3.4. *Let $\Phi = \mathcal{Q} : (A \wedge B)$ with $\mathcal{Q} = Q_1 x_1 \dots Q_n x_n$ be some SSAT formula, and the pair (c^p, I) be derivable from Φ by interpolating S-resolution, where $\mathcal{Q}(c) = Q_1 x_1 \dots Q_i x_i$. Then, for each $\tau : \text{Var}(A \wedge B) \downarrow_i \rightarrow \mathbb{B}$ with $\forall x \in \text{Var}(c) : \tau(x) = \text{ff}_c(x)$ it holds that*

- (1) $\text{Var}(I) \subseteq V_{A,B}$,
- (2) $\text{Pr}(Q_{i+1} x_{i+1} \dots Q_n x_n : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0$, and
- (3) $\text{Pr}(Q_{i+1} x_{i+1} \dots Q_n x_n : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0$.

Proof. We prove the lemma by induction over application of the interpolating S-resolution rules **RI.1**, **RI.2**, and **RI.3**. In the base case, we can just apply **RI.1** and **RI.2**. Item 1 clearly holds for both rules since I contains only variables in $V_{A,B}$. Let us consider **RI.1** first. If $c \in A$ then $I = \mathbf{false}$. By construction of τ , i.e. c evaluates to **false** under τ , it follows that $A[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]$ is unsatisfiable and thus

$$\text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0.$$

As $I = \mathbf{false}$, immediately

$$\text{Pr}(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0.$$

If $c \in B$ then $I = \mathbf{true}$. Obviously,

$$\text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0$$

and by construction of τ ,

$$\text{Pr}(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0.$$

For rule **RI.2**, we have $\models (A \wedge B)[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]$ which immediately implies that $\models (\exists a_1, \dots, a_\alpha, b_1, \dots, b_\beta : (A \wedge B))[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]$, i.e. $\models \mathcal{S}_{A,B}[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]$ by definition of $\mathcal{S}_{A,B}$. Rephrasing the latter, $\neg \mathcal{S}_{A,B}[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]$ is unsatisfiable. Consequently, for any propositional formula I

$$\text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0,$$

$$\text{Pr}(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0.$$

This proves items 2 and 3 for the base case.

In the induction step, we now assume that the lemma holds for all clauses in the premises of rule **RI.3**. Then, by construction of I , item 1 clearly holds for I , i.e. $\text{Var}(I) \subseteq V_{A,B}$. Induction hypothesis assumes that

$$\text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_1)[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\mathbf{true}/x_j]) = 0,$$

$$\text{Pr}(\mathcal{Q}' : (I_1 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\mathbf{true}/x_j]) = 0$$

holds for $((c_1 \vee \neg x_j)^{p_1}, I_1)$ and for each $\tau_1 : \text{Var}(A \wedge B) \downarrow_{j-1} \rightarrow \mathbb{B}$ with $\forall x \in \text{Var}(c_1) : \tau_1(x) = \text{ff}_{c_1}(x)$, and that

$$\text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_2)[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\text{false}/x_j]) = 0 ,$$

$$\text{Pr}(\mathcal{Q}' : (I_2 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\text{false}/x_j]) = 0$$

holds for $((c_2 \vee x_j)^{p_2}, I_2)$ and for each $\tau_2 : \text{Var}(A \wedge B) \downarrow_{j-1} \rightarrow \mathbb{B}$ with $\forall x \in \text{Var}(c_2) : \tau_2(x) = \text{ff}_{c_2}(x)$, where $j \geq i+1$ and $\mathcal{Q}' = Q_{j+1}x_{j+1} \dots Q_n x_n$. Let $\tau : \text{Var}(A \wedge B) \downarrow_{j-1} \rightarrow \mathbb{B}$ be any assignment with $\tau(x) = \tau_1(x)$ if $x \in \text{Var}(c_1)$ and $\tau(x) = \tau_2(x)$ if $x \in \text{Var}(c_2)$. Note that τ is well-defined as $\not\models (c_1 \vee c_2)$, i.e. for each $x \in \text{Var}(c_1) \cap \text{Var}(c_2) : \tau_1(x) = \tau_2(x)$. We now show that

$$\text{Pr}_A := \text{Pr}(Q_j x_j \mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) = 0 ,$$

$$\text{Pr}_B := \text{Pr}(Q_j x_j \mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) = 0$$

by proving that

$$\text{Pr}_{A,x} := \text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}][\text{true}/x_j]) = 0 ,$$

$$\text{Pr}_{A,\neg x} := \text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}][\text{false}/x_j]) = 0 ,$$

$$\text{Pr}_{B,x} := \text{Pr}(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}][\text{true}/x_j]) = 0 ,$$

$$\text{Pr}_{B,\neg x} := \text{Pr}(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}][\text{false}/x_j]) = 0 .$$

We therefore distinguish the three cases $x_j \in V_A$, $x_j \in V_B$, and $x_j \in V_{A,B}$.

First, let be $x_j \in V_A$. Then, $I = I_1 \vee I_2$. By induction hypothesis and by construction of I ,

$$\begin{aligned} 0 &= \text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_1)[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\text{true}/x_j]) \\ &\geq \text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_1 \wedge \neg I_2)[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\text{true}/x_j]) \\ &= \text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\text{true}/x_j]) . \end{aligned}$$

Due to construction of τ , it holds in particular that

$$0 = \text{Pr}_{A,x} .$$

Analogously,

$$\begin{aligned} 0 &= \text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_2)[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\text{false}/x_j]) \\ &\geq \text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_1 \wedge \neg I_2)[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\text{false}/x_j]) \\ &= \text{Pr}(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\text{false}/x_j]) \end{aligned}$$

and thus

$$0 = \text{Pr}_{A,\neg x} .$$

As $x_j \notin \text{Var}(I) \cup \text{Var}(B) \cup \text{Var}(\neg \mathcal{S}_{A,B})$, for each $v \in \mathbb{B}$ it holds that

$$\begin{aligned} &\text{Pr}(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}][v/x_j]) \\ &= \text{Pr}(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) \end{aligned}$$

which implies $\text{Pr}_{B,x} = \text{Pr}_{B,\neg x}$. We conclude from induction hypothesis that

$$\text{Pr}(\mathcal{Q}' : (I_1 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}]) = 0 ,$$

$$Pr(\mathcal{Q}' : (I_2 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}]) = 0$$

again by virtue of $x_j \notin \text{Var}(I) \cup \text{Var}(B) \cup \text{Var}(\neg \mathcal{S}_{A,B})$. Moreover,

$$Pr(\mathcal{Q}' : (I_1 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) = 0 ,$$

$$Pr(\mathcal{Q}' : (I_2 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) = 0$$

due to construction of τ . Note that if $Pr(\mathcal{Q} : \varphi_1) = 0$ and $Pr(\mathcal{Q} : \varphi_2) = 0$ then $Pr(\mathcal{Q} : (\varphi_1 \vee \varphi_2)) = 0$ since $Pr(\mathcal{Q} : \varphi) = 0$ if and only if φ is unsatisfiable.⁶ As a consequence,

$$\begin{aligned} 0 &= Pr \left(\mathcal{Q}' : \left(\begin{array}{c} (I_1 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}] \\ \vee \\ (I_2 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}] \end{array} \right) \right) \\ &= Pr(\mathcal{Q}' : ((I_1 \vee I_2) \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) \\ &= Pr(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) \\ &= Pr_{B,x} = Pr_{B,\neg x} . \end{aligned}$$

Second, let be $x_j \in V_B$. Then, $I = I_1 \wedge I_2$. As $x_j \notin \text{Var}(A) \cup \text{Var}(\neg \mathcal{S}_{A,B}) \cup \text{Var}(\neg I)$, with the same argument as above,

$$\begin{aligned} 0 &= Pr \left(\mathcal{Q}' : \left(\begin{array}{c} (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_1)[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}] \\ \vee \\ (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_2)[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}] \end{array} \right) \right) \\ &= Pr(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge (\neg I_1 \vee \neg I_2))[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) \\ &= Pr(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) \\ &= Pr_{A,x} = Pr_{A,\neg x} . \end{aligned}$$

Again following the reasoning above, we have

$$\begin{aligned} 0 &= Pr(\mathcal{Q}' : (I_1 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\mathbf{true}/x_j]) \\ &\geq Pr(\mathcal{Q}' : (I_1 \wedge I_2 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\mathbf{true}/x_j]) \\ &= Pr(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\mathbf{true}/x_j]) \end{aligned}$$

and thus

$$0 = Pr_{B,x}$$

as well as

$$\begin{aligned} 0 &= Pr(\mathcal{Q}' : (I_2 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\mathbf{false}/x_j]) \\ &\geq Pr(\mathcal{Q}' : (I_1 \wedge I_2 \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\mathbf{false}/x_j]) \\ &= Pr(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}][\mathbf{false}/x_j]) , \end{aligned}$$

and thus

$$0 = Pr_{B,\neg x} .$$

Third, let be $x_j \in V_{A,B}$. Then, $I = (\neg x_j \vee I_1) \wedge (x_j \vee I_2)$, and we deduce

$$0 = Pr(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_1)[\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}][\mathbf{true}/x_j])$$

⁶This statement is not true in general if \mathcal{Q} also contains *universal* quantifiers, which is not the case in this article. However, extensions of SSAT involving universal quantifiers have also been considered in the literature, confer [Maj09].

$$\begin{aligned}
&= Pr(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \\
&\quad \wedge ((x_j \wedge \neg I_1) \vee (\neg x_j \wedge \neg I_2))) [\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}] [\mathbf{true}/x_j]) \\
&= Pr(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I) [\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}] [\mathbf{true}/x_j])
\end{aligned}$$

and, in particular,

$$0 = Pr_{A,x} .$$

Analogously,

$$\begin{aligned}
0 &= Pr(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I_2) [\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}] [\mathbf{false}/x_j]) \\
&= Pr(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \\
&\quad \wedge ((x_j \wedge \neg I_1) \vee (\neg x_j \wedge \neg I_2))) [\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}] [\mathbf{false}/x_j]) \\
&= Pr(\mathcal{Q}' : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I) [\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}] [\mathbf{false}/x_j])
\end{aligned}$$

and, in particular,

$$0 = Pr_{A,\neg x} .$$

Furthermore,

$$\begin{aligned}
0 &= Pr(\mathcal{Q}' : (I_1 \wedge B \wedge \neg \mathcal{S}_{A,B}) [\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}] [\mathbf{true}/x_j]) \\
&= Pr(\mathcal{Q}' : ((\neg x_j \vee I_1) \wedge (x_j \vee I_2) \wedge B \\
&\quad \wedge \neg \mathcal{S}_{A,B}) [\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}] [\mathbf{true}/x_j]) \\
&= Pr(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B}) [\tau_1(x_1)/x_1] \dots [\tau_1(x_{j-1})/x_{j-1}] [\mathbf{true}/x_j])
\end{aligned}$$

and, in particular,

$$0 = Pr_{B,x} .$$

Finally,

$$\begin{aligned}
0 &= Pr(\mathcal{Q}' : (I_2 \wedge B \wedge \neg \mathcal{S}_{A,B}) [\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}] [\mathbf{false}/x_j]) \\
&= Pr(\mathcal{Q}' : ((\neg x_j \vee I_1) \wedge (x_j \vee I_2) \wedge B \\
&\quad \wedge \neg \mathcal{S}_{A,B}) [\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}] [\mathbf{false}/x_j]) \\
&= Pr(\mathcal{Q}' : (I \wedge B \wedge \neg \mathcal{S}_{A,B}) [\tau_2(x_1)/x_1] \dots [\tau_2(x_{j-1})/x_{j-1}] [\mathbf{false}/x_j])
\end{aligned}$$

and, in particular,

$$0 = Pr_{B,\neg x} .$$

Having shown that $Pr_{A,x} = Pr_{A,\neg x} = Pr_{B,x} = Pr_{B,\neg x} = 0$, we can now prove the intermediate result above, i.e. $Pr_A = Pr_B = 0$. If $Q_j = \exists$ then $Pr_A = \max(Pr_{A,x}, Pr_{A,\neg x}) = 0$ and $Pr_B = \max(Pr_{B,x}, Pr_{B,\neg x}) = 0$, and if $Q_j = \forall^{p_x}$ then $Pr_A = p_x \cdot Pr_{A,x} + (1-p_x) \cdot Pr_{A,\neg x} = 0$ and $Pr_B = p_x \cdot Pr_{B,x} + (1-p_x) \cdot Pr_{B,\neg x} = 0$.

To finish the proof, we finally need to show that items 2 and 3, i.e.

$$Pr(Q_{i+1}x_{i+1} \dots Q_nx_n : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I) [\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0 ,$$

$$Pr(Q_{i+1}x_{i+1} \dots Q_n x_n : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0 ,$$

follow from $Pr_A = Pr_B = 0$, i.e. from

$$Pr(Q_j x_j \dots Q_n x_n : (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) = 0 ,$$

$$Pr(Q_j x_j \dots Q_n x_n : (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{j-1})/x_{j-1}]) = 0 .$$

If $j = i + 1$ then the result is obvious. Otherwise, i.e. if $j > i + 1$, the variables x_{i+1}, \dots, x_{j-1} do not occur in the derived clause $(c_1 \vee c_2)$ since $\mathcal{Q}(c_1 \vee c_2) = Q_1 x_1 \dots Q_i x_i$. By definition of assignment τ , for $k = j - 1$ down to $i + 1$ we may therefore successively conclude that

$$\begin{aligned} Pr(Q_{k+1}x_{k+1} \dots Q_n x_n : \\ (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_{k-1})/x_{k-1}][\mathbf{true}/x_k]) &= 0 , \end{aligned}$$

$$\begin{aligned} Pr(Q_{k+1}x_{k+1} \dots Q_n x_n : \\ (A \wedge \neg \mathcal{S}_{A,B} \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_{k-1})/x_{k-1}][\mathbf{false}/x_k]) &= 0 , \end{aligned}$$

$$\begin{aligned} Pr(Q_{k+1}x_{k+1} \dots Q_n x_n : \\ (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{k-1})/x_{k-1}][\mathbf{true}/x_k]) &= 0 , \end{aligned}$$

$$\begin{aligned} Pr(Q_{k+1}x_{k+1} \dots Q_n x_n : \\ (I \wedge B \wedge \neg \mathcal{S}_{A,B})[\tau(x_1)/x_1] \dots [\tau(x_{k-1})/x_{k-1}][\mathbf{false}/x_k]) &= 0 . \end{aligned}$$

From case $k = i + 1$ the result immediately follows. \square

Completeness of S-resolution, as stated in Theorem 3.3, together with above Lemma 3.4, applied to the derived pair (\emptyset^p, I) , yields

Corollary 3.5 (Generalized Craig interpolants computation). *If $\mathcal{Q} : (A \wedge B)$ is an SSAT formula then a generalized Craig interpolant for (A, B) can be computed by interpolating S-resolution.* \square

Note that computation of generalized interpolants does not depend on the actual truth state of $A \wedge B$. The next observation facilitates to effectively control the geometric extent of generalized Craig interpolants within the “don’t care”-region $\mathcal{S}_{A,B}$. This result will be useful within applications of generalized Craig interpolation to the symbolic analysis of probabilistic systems being investigated in Section 4.

Corollary 3.6 (Controlling generalized Craig interpolants computation). *If $I = \mathbf{true}$ is used within each application of rule **RI.2** then $Pr(\mathcal{Q} : (A \wedge \neg \mathcal{I})) = 0$. Likewise, if $I = \mathbf{false}$ is used in rule **RI.2** then $Pr(\mathcal{Q} : (\mathcal{I} \wedge B)) = 0$.*

Proof. The proof works analogously to the one of Lemma 3.4. For the base case, it is clear that the desired property for **RI.1** is independent of $\neg \mathcal{S}_{A,B}$. For **RI.2**, if $I = \mathbf{true}$ then clearly $Pr(\mathcal{Q}' : (A \wedge \neg I)[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0$, and if $I = \mathbf{false}$ then $Pr(\mathcal{Q}' : (I \wedge B)[\tau(x_1)/x_1] \dots [\tau(x_i)/x_i]) = 0$. Then, we can modify the induction hypothesis: for case “ $I = \mathbf{true}$ in **RI.2**”, we assume that $Pr(\mathcal{Q}' : (A \wedge \neg I_1)[\tau_1(x_1)/x_1] \dots [\mathbf{true}/x_j]) = 0$, $Pr(\mathcal{Q}' : (A \wedge \neg I_2)[\tau_2(x_1)/x_1] \dots [\mathbf{false}/x_j]) = 0$, and for “ $I = \mathbf{false}$ in **RI.2**” that $Pr(\mathcal{Q}' : (I_1 \wedge B)[\tau_1(x_1)/x_1] \dots [\mathbf{true}/x_j]) = 0$, $Pr(\mathcal{Q}' : (I_2 \wedge B)[\tau_2(x_1)/x_1] \dots [\mathbf{false}/x_j]) = 0$. The induction step then follows the same reasoning as in the remaining proof of Lemma 3.4. \square

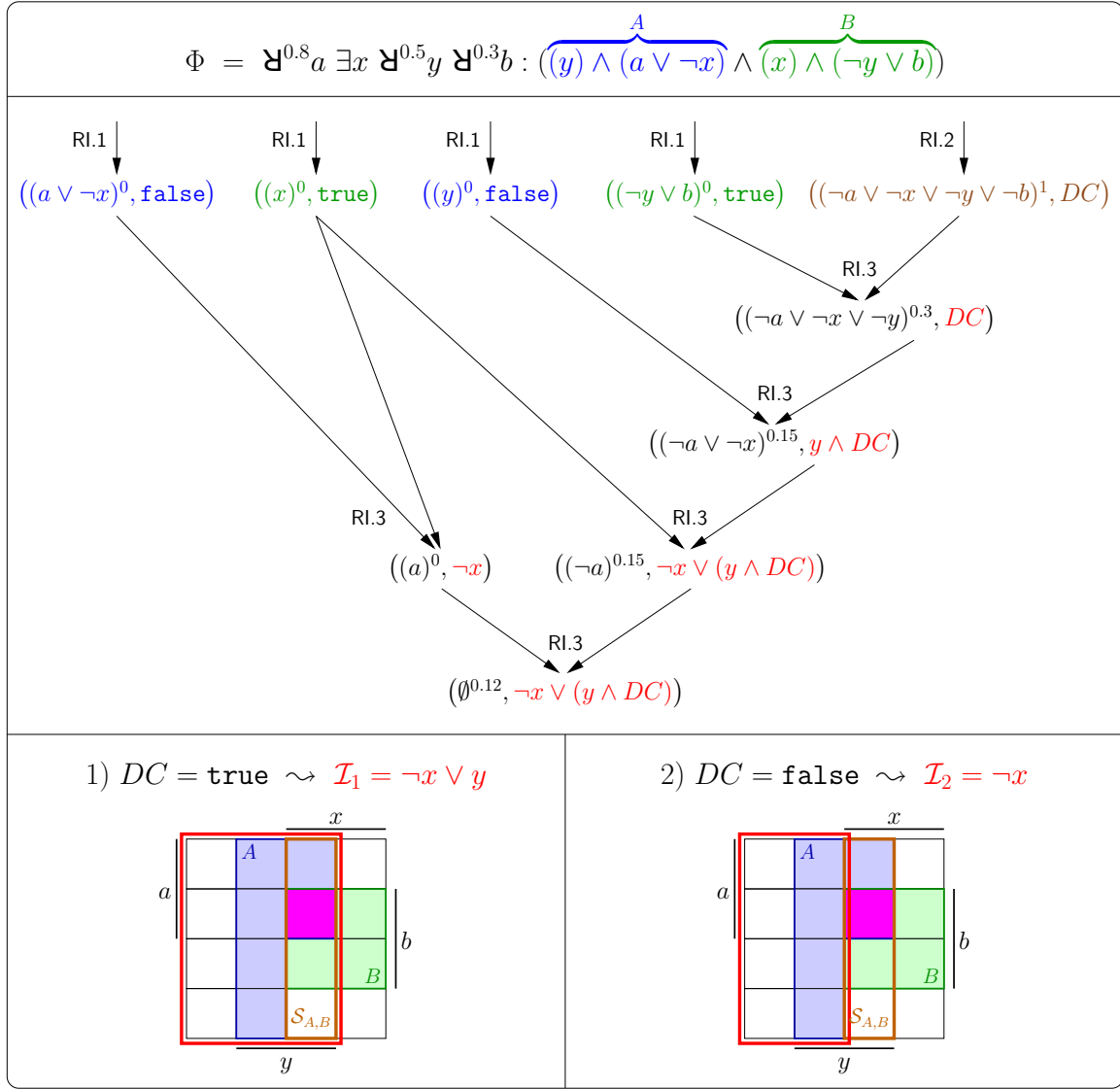
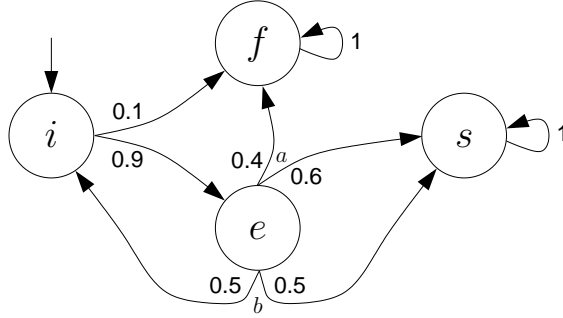


Figure 3: Example of interpolating S-resolution and illustration of the resulting generalized Craig interpolants by means of Karnaugh-Veitch diagrams. Arrows denote applications of the specified interpolating S-resolution rules, while DC stands for any formula over $V_{A,B}$ as in rule **RI.2**.

Observe that the special interpolants \mathcal{I} from Corollary 3.6 relate to the classical strongest and weakest Craig interpolants A^\exists and \overline{B}^\forall , respectively, in the following sense: $Pr(\mathcal{Q} : (A \wedge \neg \mathcal{I})) = 0$ iff $\models A \Rightarrow \mathcal{I}$ iff $\models \forall a_1, \dots, a_\alpha : (A \Rightarrow \mathcal{I})$ iff $\models (A^\exists \Rightarrow \mathcal{I})$, as a_1, \dots, a_α do not occur in \mathcal{I} . Analogously, $Pr(\mathcal{Q} : (\mathcal{I} \wedge B)) = 0$ iff $\models \mathcal{I} \Rightarrow \neg B$ iff $\models \forall b_1, \dots, b_\beta : (\mathcal{I} \Rightarrow \neg B)$ iff $\models \mathcal{I} \Rightarrow \overline{B}^\forall$.

Figure 4: A simple MDP \mathcal{M} .

Example of computing generalized Craig interpolants by interpolating S-resolution. For an example of interpolating S-resolution, consider the SSAT formula $\Phi = \mathfrak{A}^{0.8}a \exists x \mathfrak{A}^{0.5}y \mathfrak{A}^{0.3}b : (A \wedge B)$ with $A = ((y) \wedge (a \vee \neg x))$ and $B = ((x) \wedge (\neg y \vee b))$. Then, $V_A = \{a\}$, $V_B = \{b\}$, and $V_{A,B} = \{x, y\}$. It is not hard to see that the only satisfying assignment τ of the propositional formula $A \wedge B$ is given by $\tau(a) = \text{true}$, $\tau(x) = \text{true}$, $\tau(y) = \text{true}$, and $\tau(b) = \text{true}$. Hence, $Pr(\Phi) = 0.12$. A derivation of the empty clause $\emptyset^{0.12}$ together with its associated generalized Craig interpolant $\neg x \vee (y \wedge DC)$ is shown in Figure 3, while DC stands for any formula over variables in $V_{A,B}$ as in rule **R1.2**. Note that pair $((\neg a \vee \neg x \vee \neg y \vee \neg b)^1, DC)$ is derivable by rule **R1.2** since $\models (A \wedge B)[\tau(a)/a][\tau(x)/x][\tau(y)/y][\tau(b)/b]$. Applying Corollary 3.6 by choosing $DC = \text{true}$ and $DC = \text{false}$, we obtain the generalized Craig interpolants $\mathcal{I}_1 = \neg x \vee y$ and $\mathcal{I}_2 = \neg x$, respectively, such that $Pr(\mathcal{Q} : (A \wedge \neg \mathcal{I}_1)) = 0$ and $Pr(\mathcal{Q} : (\mathcal{I}_2 \wedge B)) = 0$. In other words, $A \Rightarrow \mathcal{I}_1$ and $\mathcal{I}_2 \Rightarrow \neg B$, as illustrated by the Karnaugh-Veitch diagrams in Figure 3.

4. APPLICATIONS OF GENERALIZED CRAIG INTERPOLATION TO ANALYSIS OF PROBABILISTIC SYSTEMS

In this section, we investigate the application of generalized Craig interpolation to the symbolic analysis of probabilistic systems. We direct our attention to two analysis goals, namely to *probabilistic state reachability* in Section 4.1 as well as to *probabilistic region stability* in Section 4.2. As a system model, we consider finite-state Markov decision processes (MDPs) [Bel57]. An MDP $\mathcal{M} = (\iota, S, Act, ps(\cdot, \cdot, \cdot))$ is a finite-state system in which state changes are subject to *non-deterministic* selection among available actions followed by a *probabilistic* choice among potential successor states, while the probability distribution of the latter choice depends on the selected action. More precisely, S is a finite set of states, $\iota \in S$ is the initial state, Act is a finite set of actions, and $ps(s, a, s')$ gives the probability that \mathcal{M} performs a transition step from $s \in S$ to $s' \in S$ under action $a \in Act$. For an example, consider the simple MDP \mathcal{M} from Figure 4 where $\iota = i$, $S = \{i, f, e, s\}$, and $Act = \{a, b\}$. A transition $ps(z, act, z') = p > 0$ is indicated by an arrow from z to z' accompanied by action act and by the corresponding transition probability p . If two states are not connected by an arrow then the corresponding transition probability is 0, and if no action is specified then that transition is feasible for all actions. A probability measure of an MDP is well-defined only if considering a particular scheduler σ resolving the non-determinism. That is, σ schedules the action for the current state. Different such

schedulers σ have been investigated in the literature, confer, for instance, [BHKH05]: σ may select the next action either in a deterministic or randomized fashion. In both cases, σ may have access to and thus base its selection on either the current state only or the full system history. In our scenarios, we do not manipulate schedulers explicitly, but define the probability measures obtained by worst-case deterministic schedulers achieving maximum or minimum, depending on how the worst case is understood, probability of reaching target states directly as the limit of a recursive function over \mathbb{N} . For each $k \in \mathbb{N}$, the recursive function determines the maximum or minimum probability of reaching target states within k steps, as achieved by a worst-case history-dependent scheduler. As a worst-case history-dependent scheduler will always maximize or minimize the probability of reaching the target within the remaining number of steps, its performance coincides with the probabilities computed by a backward induction resolving non-deterministic choices by taking the maximum or minimum, respectively, of the probability values obtained from the next-lower recursion depth.

All experiments mentioned in this section were performed on a 1.83 GHz Intel Core 2 Duo machine with 1 GByte physical memory running Linux.

4.1. Interpolation-based probabilistic state reachability. Let be given an MDP \mathcal{M} and a set of target states $Target \subseteq S$ in \mathcal{M} . With regard to *probabilistic state reachability*, the goal is to compute the probability of reaching the target states $Target$ from the initial state ι under some explicitly or implicitly (e.g., by an optimality condition) given scheduler σ . In most applications, the target states are considered to be *bad*, for instance, to be fatal system errors, such that one is faced with computing the *worst-case* probability of reaching the bad states, i.e. *maximizing* the reachability probability under each possible scheduler. This maximum probability $MaxReach(\mathcal{M}, Target)$ can be defined directly as the limit of the maximum step-bounded probability of reaching the target states as similarly shown by [FHH⁺11, Lemma 1], i.e.

$$MaxReach(\mathcal{M}, Target) = \lim_{k \rightarrow \infty} MaxReach_{\mathcal{M}, Target}^k(\iota)$$

where

$$MaxReach_{\mathcal{M}, Target}^k(s) = \begin{cases} 1 & ; s \in Target \\ 0 & ; s \notin Target, k = 0 \\ \max_{a \in Act} \sum_{s' \in S} ps(s, a, s') \cdot MaxReach_{\mathcal{M}, Target}^{k-1}(s') & ; s \notin Target, k > 0 \end{cases}$$

gives the maximum probability of reaching the target states from state $s \in S$ within k steps ($k \in \mathbb{N}$) under each possible scheduler. For some threshold value $\theta \in [0, 1]$, the *safety verification problem* is to decide whether the worst-case probability of reaching the bad states is at most θ , i.e. to decide whether

$$MaxReach(\mathcal{M}, Target) \leq \theta \tag{4.1}$$

holds.

In previous work [FHT08, FTE10, TEF11], we have established a symbolic *falsification* procedure for above problem 4.1. Though this approach is based on SSMT, i.e. an arithmetic extension of SSAT, and works for the more general class of discrete-time probabilistic hybrid systems, which roughly are MDPs with arithmetic-logical transition guards and actions, the same procedure restricted to SSAT is applicable for finite-state MDPs. The key idea here is to adapt *bounded model checking* (BMC) [BCCZ99] to the probabilistic case by

encoding step-bounded reachability as an SSAT problem: like in classical BMC, the initial states, the transition relation, and the target states of an MDP \mathcal{M} are symbolically encoded by propositional formulae in CNF, namely by $Init(\mathbf{s})$, $Trans(\mathbf{s}, \mathbf{nt}, \mathbf{pt}, \mathbf{s}')$, and $Target(\mathbf{s})$, respectively, where the propositional variable vector \mathbf{s} represents the system state before and \mathbf{s}' after a transition step. To keep track of the *non-deterministic* and *probabilistic* selections of transitions in $Trans(\mathbf{s}, \mathbf{nt}, \mathbf{pt}, \mathbf{s}')$, we further introduce propositional variables \mathbf{nt} and \mathbf{pt} to encode non-deterministic selection among available actions and to describe probabilistic choice of the successor state, respectively. Assignments to these variables determine which of possibly multiple available transitions departing from \mathbf{s} is taken. In contrast to traditional BMC, all variables are quantified: all state variables \mathbf{s} and \mathbf{s}' are existentially quantified in the prefixes $\mathcal{Q}_{\mathbf{s}}$ and $\mathcal{Q}_{\mathbf{s}'}$. The transition-selection variables \mathbf{nt} encoding non-deterministic choice are *existentially quantified* by $\mathcal{Q}_{\mathbf{nt}}$, while the probabilistic selector variables \mathbf{pt} are bound by *randomized quantifiers* in $\mathcal{Q}_{\mathbf{pt}}$.⁷ For the sake of clarity, let be $\mathbf{t} := \mathbf{nt} \cup \mathbf{pt}$ and $\mathcal{Q}_{\mathbf{t}} := \mathcal{Q}_{\mathbf{nt}} \mathcal{Q}_{\mathbf{pt}}$.

According to [FHT08, Proposition 1], the maximum probability of reaching the target states in \mathcal{M} from the initial state ι within k transition steps, i.e. $MaxReach_{\mathcal{M}, Target}^k(\iota)$, is equal to the satisfaction probability

$$lb_k := Pr\left(\mathcal{Q}(k) : \left(\overbrace{Init(\mathbf{s}_0) \wedge \bigwedge_{i=1}^k Trans(\mathbf{s}_{i-1}, \mathbf{t}_i, \mathbf{s}_i)}^{\text{states reachable within } k \text{ steps}} \wedge \left(\overbrace{\bigvee_{i=0}^k Target(\mathbf{s}_i)}^{\text{hit target states}} \right) \right) \right) \quad (4.2)$$

with $\mathcal{Q}(k) := \mathcal{Q}_{\mathbf{s}_0} \mathcal{Q}_{\mathbf{t}_1} \mathcal{Q}_{\mathbf{s}_1} \dots \mathcal{Q}_{\mathbf{s}_{k-1}} \mathcal{Q}_{\mathbf{t}_k} \mathcal{Q}_{\mathbf{s}_k}$.

Observe that each value $lb_k = MaxReach_{\mathcal{M}, Target}^k(\iota)$ can be computed by an SSAT solver and constitutes a lower bound of the maximum reachability probability $MaxReach(\mathcal{M}, Target)$ due to monotonicity of the chain $(MaxReach_{\mathcal{M}, Target}^k(\iota))_{k \in \mathbb{N}}$. This symbolic approach, called *probabilistic bounded model checking* (PBMC), is able to *falsify* safety properties of shape 4.1 once a value $lb_k > \theta$ is computed for some k .

However, the development of a corresponding counterpart based on SSAT that is able to compute *upper* bounds ub_k of the maximum reachability probability $MaxReach(\mathcal{M}, Target)$ was left as an open challenge. Such an approach would permit to *verify* safety properties of shape 4.1 once a value $ub_k \leq \theta$ is computed for some k .

In the remainder of this section, we propose such a symbolic *verification* procedure for above problem 4.1 by means of generalized Craig interpolation. This verification method proceeds in two phases. Phase 1 computes a symbolic representation of an *overapproximation of the backward reachable state set*, where a state is backward reachable if it is the origin of a transition sequence leading into *Target*. Phase 1 can be integrated into PBMC, as used to falsify the probabilistic safety property. Whenever such falsification fails for a given step depth k , we apply generalized Craig interpolation to the (just failed) PBMC proof to compute a *symbolic overapproximation of the backward reachable state set* at depth k and then proceed to PBMC at some higher depth $k' > k$. As an alternative to the integration into PBMC, interpolants describing the backward reachable state sets can be successively extended by “stepping” them by prepending another transition, as explained below. In either case, phase 1 ends when the backward reachable state set becomes stable, in which case we

⁷Non-deterministic branching of n alternatives can be represented by a binary tree of depth $\lceil \log_2 n \rceil$ and probabilistic branching by a sequence of at most $n - 1$ binary branches, yielding $\lceil \log_2 n \rceil$ existential and $n - 1$ randomized quantifiers, respectively.

have computed a symbolic overapproximation of the whole backward reachable state set. In phase 2, we construct an SSAT formula with parameter k that forces the system to *stay within the backward reachable state set* for k steps. The maximum satisfaction probability of that SSAT formula then gives an upper bound on the maximum probability of reaching the target states. The rationale is that system runs leaving the backward reachable state set will never reach the target states.

Phase 1. Given an SSAT encoding of an MDP \mathcal{M} as above, the state-set predicate $\mathcal{B}^k(\mathbf{s})$ for $k \in \mathbb{N}$ over state variables \mathbf{s} is inductively defined as

- $\mathcal{B}^0(\mathbf{s}) := \text{Target}(\mathbf{s})$, and
- $\mathcal{B}^{k+1}(\mathbf{s}) := \mathcal{B}^k(\mathbf{s}) \vee \mathcal{I}^{k+1}(\mathbf{s})$

where $\mathcal{I}^{k+1}(\mathbf{s}_{j-1})$ is a generalized Craig interpolant for

$$\left(\overbrace{\text{Trans}(\mathbf{s}_{j-1}, \mathbf{t}_j, \mathbf{s}_j) \wedge \mathcal{B}^k(\mathbf{s}_j)}^{=A}, \overbrace{\text{Init}(\mathbf{s}_0) \wedge \bigwedge_{i=1}^{j-1} \text{Trans}(\mathbf{s}_{i-1}, \mathbf{t}_i, \mathbf{s}_i)}^{=B} \right)$$

with $j \geq 1$ with respect to SSAT formula

$$\mathcal{Q}(j) : \left(\overbrace{\text{Init}(\mathbf{s}_0) \wedge \bigwedge_{i=1}^{j-1} \text{Trans}(\mathbf{s}_{i-1}, \mathbf{t}_i, \mathbf{s}_i)}^{j-1 \text{ steps "forward" } (=B)} \wedge \overbrace{\text{Trans}(\mathbf{s}_{j-1}, \mathbf{t}_j, \mathbf{s}_j) \wedge \mathcal{B}^k(\mathbf{s}_j)}^{\text{one step "backward" } (=A)} \right). \quad (4.3)$$

Observe that each generalized Craig interpolant $\mathcal{I}^{k+1}(\mathbf{s})$ can be computed by interpolating S-resolution if we rewrite $\mathcal{B}^k(\mathbf{s})$ into CNF, the latter being always possible in linear time by adding auxiliary V_A -variables. During computation of each $\mathcal{I}^{k+1}(\mathbf{s})$, we take $I = \mathbf{true}$ in every application of rule **RI.2** such that $\mathcal{B}^k(\mathbf{s})$ overapproximates all system states backward reachable from target states within k steps due to Corollary 3.6. Whenever $\mathcal{B}^k(\mathbf{s})$ has stabilized, i.e.

$$\mathcal{B}^{k+1}(\mathbf{s}) \Rightarrow \mathcal{B}^k(\mathbf{s}),$$

we can be sure that $\mathcal{B}(\mathbf{s}) := \mathcal{B}^k(\mathbf{s})$ overapproximates all backward reachable states. It is obvious that $\mathcal{B}^k(\mathbf{s})$ finally stabilizes in the finite-state case.

Note that parameter $j \geq 1$ can be chosen arbitrarily, i.e. the system may execute any number of transitions until state \mathbf{s}_{j-1} is reached since this does not destroy the “backward-overapproximating” property of $\mathcal{B}^{k+1}(\mathbf{s})$. The rationale of having parameter j is the additional freedom in constructing generalized interpolants since j may influence the shape of $\mathcal{I}^{k+1}(\mathbf{s})$, as we will see in the example below.

We remark that phase 1 is a clean generalization of McMillan’s approach [McM03, McM05], the latter having unsatisfiability of $A \wedge B$ as precondition in each iteration k .⁸

⁸Instead of overapproximating the backward reachable state set, McMillan’s scheme [McM03, McM05] actually targets at forward reachable states, which however makes no fundamental difference in the non-probabilistic setting.

Phase 2. Having symbolically described all backward reachable states by the predicate $\mathcal{B}(\mathbf{s})$, upper bounds ub_k of the maximum probability $MaxReach(\mathcal{M}, Target)$ of reaching the target states $Target$ can now be computed by SSAT solving applied to

$$ub_k := Pr\left(\mathcal{Q}(k) : \left(\overbrace{Init(\mathbf{s}_0) \wedge \bigwedge_{i=1}^k Trans(\mathbf{s}_{i-1}, \mathbf{t}_i, \mathbf{s}_i)}^{\text{states reachable within } k \text{ steps}} \wedge \overbrace{\bigwedge_{i=0}^k \mathcal{B}(\mathbf{s}_i)}^{\text{stay in back-reach set}} \right) \right). \quad (4.4)$$

First observe that the formula above excludes all system runs that leave the set of backward reachable states. This is sound since leaving $\mathcal{B}(\mathbf{s})$ means to never reach the $Target(\mathbf{s})$ states. Second, the system behavior becomes more and more constrained for increasing k , i.e. the ub_k 's are monotonically decreasing. With regard to solving problem 4.1, the safety property $MaxReach(\mathcal{M}, Target) \leq \theta$ is *verified* by the procedure above once an upper bound $ub_k \leq \theta$ is computed for some k .

Example. To illustrate the symbolic approach to probabilistic safety verification based on generalized Craig interpolation, consider the simple MDP \mathcal{M} from Figure 4 with s being the only target state.

With regard to the symbolic encoding of \mathcal{M} , we introduce four Boolean variables i, f, e, s to describe the state space. The literal i means that \mathcal{M} is in state i while literal $\neg i$ expresses that \mathcal{M} is not in i . The same holds analogously for the other states. Note that, in order to encode valid system states, we have to ensure that exactly one of the variables i, f, e, s is **true** in each time instant. The encoding of this constraint will be explained later on. The non-deterministic choice between actions a and b is encoded by a Boolean variable act while action a is represented by the positive literal act and action b by the negative literal $\neg act$. For the three probabilistic choices in \mathcal{M} , we introduce three Boolean variables pi for the choice from i , pea for the choice from e under action a , and peb for the choice from e under action b . Recall that all state variables as well as variables encoding non-deterministic selection are existentially quantified while variables describing probabilistic choices are bound by randomized quantifiers. We thus obtain the corresponding quantifier prefixes

$$\begin{aligned} \mathcal{Q}_s &= \exists i \exists f \exists e \exists s, \\ \mathcal{Q}_t &= \exists act \ \mathfrak{P}^{0.9} pi \ \mathfrak{P}^{0.6} pea \ \mathfrak{P}^{0.5} peb, \\ \mathcal{Q}_{s'} &= \exists i' \exists f' \exists e' \exists s'. \end{aligned}$$

The formulae in CNF representing the initial state and the target states are specified by

$$Init(\mathbf{s}) = (i) \wedge (\neg f) \wedge (\neg e) \wedge (\neg s) \text{ and } Target(\mathbf{s}) = (s),$$

respectively. To obtain the transition relation predicate, we encode each single transition step. For instance, a step from state e to f under action a can be encoded by the implication $(e \wedge act \wedge \neg pea) \Rightarrow f'$, the latter being equivalent to the clause $(\neg e \vee \neg act \vee pea \vee f')$. The conjunction of all these clauses then encodes the full system behavior symbolically. Since we represent each system state by an own Boolean variable, as mentioned above, we need to enforce that exactly one of the primed state variables, constituting the system state after the transition step, carries value **true**. This is simply achieved by the formula in CNF $exactly_one(i', f', e', s') = (i' \vee f' \vee e' \vee s') \wedge (\neg i' \vee \neg f') \wedge (\neg i' \vee \neg e') \wedge (\neg i' \vee \neg s') \wedge (\neg f' \vee$

j	\mathcal{I}^1	\mathcal{B}^1	\mathcal{I}^2	\mathcal{B}^2	\mathcal{I}^3	\mathcal{B}^3	\mathcal{B}
1	$\neg i$	$\neg i \vee s$	true	true	true	true	true
	$\{f, e, s\}$	$\{f, e, s\}$	$\{i, f, e, s\}$	$\{i, f, e, s\}$	$\{i, f, e, s\}$	$\{i, f, e, s\}$	$\{i, f, e, s\}$
2	$\neg f$	$\neg f \vee s$	$\neg f$	$\neg f \vee s$	—	—	$\neg f \vee s$
	$\{i, e, s\}$	$\{i, e, s\}$	$\{i, e, s\}$	$\{i, e, s\}$	—	—	$\{i, e, s\}$
3	$\neg i \wedge \neg f$	$\neg i \wedge \neg f \vee s$	$\neg f$	$\neg f \vee s$	$\neg f$	$\neg f \vee s$	$\neg f \vee s$
	$\{e, s\}$	$\{e, s\}$	$\{i, e, s\}$	$\{i, e, s\}$	$\{i, e, s\}$	$\{i, e, s\}$	$\{i, e, s\}$

Table 1: Experimental results of applying the generalized interpolation scheme 4.3 on \mathcal{M} from Figure 4 for different values of parameter j . In addition to the formal presentation of the predicates, the concrete state sets are given explicitly.

$\neg e') \wedge (\neg f' \vee \neg s') \wedge (\neg e' \vee \neg s')$. The transition relation predicate in CNF then is

$$\begin{aligned}
Trans(s, t, s') = & (\neg i \vee pi \vee f') \wedge (\neg i \vee \neg pi \vee e') \\
& \wedge (\neg e \vee \neg act \vee pea \vee f') \wedge (\neg e \vee \neg act \vee \neg pea \vee s') \\
& \wedge (\neg e \vee act \vee peb \vee s') \wedge (\neg e \vee act \vee \neg peb \vee i') \\
& \wedge (\neg f \vee f') \wedge (\neg s \vee s') \wedge exactly_one(i', f', e', s').
\end{aligned}$$

We are now interested in the maximum probability of reaching the target state s from the initial state i . Applying the PBMC scheme 4.2, we are only able to compute lower bounds lb_k of the maximum reachability probability, for instance, $lb_0 = lb_1 = 0$, $lb_2 = lb_3 = 0.54$, $lb_4 = lb_5 = 0.693$, \dots , $lb_{20} = 0.817971$, \dots , $lb_{100} = 0.818181818181803208$. The latter results were achieved by employing the SSMT solver SiSAT⁹ [TEF11] that provides a convenient input language for specifying probabilistic transition systems like MDPs. Unwinding of the system's transition relation for increasing step bounds k , i.e. the construction of the SSAT formulae specified by scheme 4.2 in our context, is done fully automatically. Furthermore, several algorithmic optimizations are exploited to improve performance of the tool. Concerning runtime, all 100 SSAT formulae were solved within 37.05 seconds, while computation of the first 20 lower bounds lb_0 to lb_{20} just needed 370 milliseconds. The highest computation time for a single SSAT problem was obtained for lb_{100} , namely 1.14 seconds. The evolution of the lb_k 's up to $k = 20$ is presented graphically on the right of Figure 5. Given these results, one can suppose that the lower bounds converge to and never exceed value $9/11 = 0.8\bar{1}$. However, there is no mathematical guarantee for the latter guess.

To overcome this limitation, we first apply the generalized interpolation scheme 4.3 to compute an overapproximation of the backward reachable state set. The latter then facilitates to compute upper bounds ub_k of the maximum reachability probability by means of scheme 4.4. In order to compute the generalized Craig interpolants $\mathcal{I}^{k+1}(s_{j-1})$ automatically during solving the SSAT formulae 4.3, we have implemented a simple DPLL-based SSAT solver that integrates interpolating S-resolution. As mentioned earlier, scheme 4.3 allows freedom in choosing parameter $j \geq 1$. This parameter permits to specify the number $j - 1$ of transition steps until system state s_{j-1} is reached, which is the common state of formula parts A and B . The experimental results of applying the generalized interpolation scheme 4.3 on the MDP \mathcal{M} for different values of j are shown in Table 1.

⁹The SiSAT tool is available on <http://sisat.gforge.avacs.org/>.

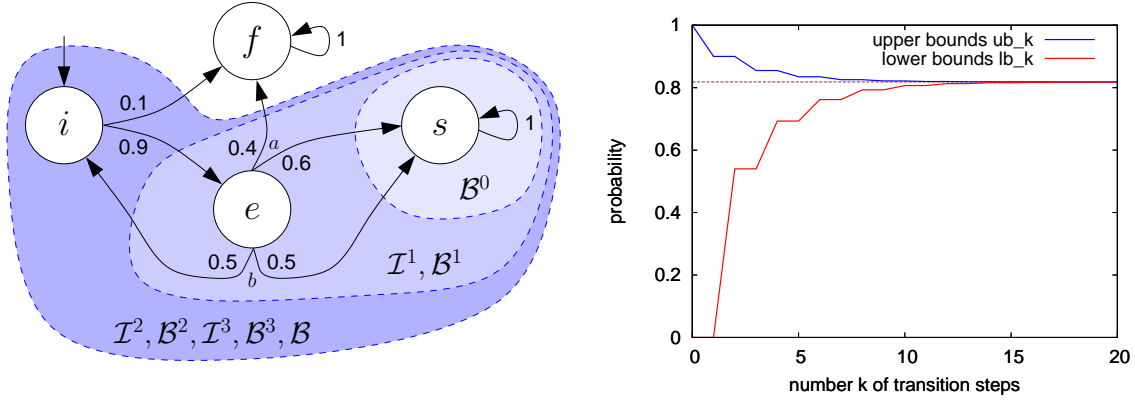


Figure 5: Illustration of the computed state sets for MDP \mathcal{M} by the generalized interpolation scheme 4.3 with $j = 3$ (left), and lower bounds lb_k and upper bounds ub_k of the maximum probability of reaching target state s over number k of transition steps computed by schemes 4.2 and 4.4, respectively (right).

From the results of Table 1, we observe that the value of j actually has an impact on the shape of the resulting interpolants. Let us consider the first interpolants \mathcal{I}^1 which overapproximate all states backward reachable in one step. Clearly, the exact set of states backward reachable in one step is $\{e, s\}$. For $j = 1$, the overapproximated set $\{f, e, s\}$ computed by the procedure is too coarse and actually contains a state which is *not* backward reachable at all, namely f . Though the set $\{i, e, s\}$ for $j = 2$ actually consists of backward reachable states only, it is not tight enough as the initial state i is backward reachable after two steps only. For $j = 3$, we achieved the precise set $\{e, s\}$. Continuing the scheme for $j = 1$, \mathcal{I}^2 and then \mathcal{I}^3 become **true** meaning that the overapproximated set of the backward reachable states \mathcal{B} covers the whole state space. Using this inconclusive result in scheme 4.4 yields only trivial upper bounds $ub_k = 1$ for all k . With regard to $j = 2$, the interpolation process has stabilized after computation of \mathcal{I}^2 . The resulting state set $\{i, e, s\}$ encoded by \mathcal{B} actually is the precise set of all backward reachable states. Though \mathcal{I}^1 was too coarse, this could be compensated in the computation of \mathcal{I}^2 . For $j = 3$, we observe that all generalized interpolants \mathcal{I}^1 , \mathcal{I}^2 , and \mathcal{I}^3 describe the corresponding backward reachable states accurately, thus leading to the precise set of all backward reachable states. The computed state sets for $j = 3$ are illustrated on the left of Figure 5. After having examined the results above, it seems that the greater the value of j , i.e. the more transition steps are performed, the more accurate the resulting overapproximation of the backward reachable state set.

Concerning runtime, each generalized Craig interpolant was computed by the interpolating DPLL-based SSAT solver within fractions of a second, where the highest runtime of 36 milliseconds was observed when computing \mathcal{I}^3 for $j = 3$.

Having computed a symbolic representation $\mathcal{B}(s)$ of an overapproximation of all backward reachable states, we are now able to compute upper bounds ub_k of the maximum reachability probability by means of scheme 4.4, where we use $\mathcal{B}(s) = \neg f \vee s$ as obtained for $j = 3$ as well as for $j = 2$. Again employing the SSMT tool SiSAT, some of the results are $ub_0 = 1$, $ub_1 = ub_2 = 0.9$, $ub_3 = ub_4 = 0.855$, $ub_5 = ub_6 = 0.83475$, ..., $ub_{20} = 0.818243$, ..., $ub_{100} = 0.81818181818181821948$. Concerning runtime, all 100 SSAT formulae were

solved within 54.76 seconds, while computation of the first 20 upper bounds ub_0 to ub_{20} just needed 400 milliseconds. The highest computation time for a single SSAT problem was obtained for ub_{100} , namely 1.77 seconds. The evolution of the ub_k 's up to $k = 20$ is presented graphically on the right of Figure 5.

In addition to estimating the maximum reachability probability from below using the PBMC scheme 4.2, we are now able to estimate the probability also from above. In our example, we can *safely* conclude that

$$0.818181818181803208 = lb_{100} \leq \text{MaxReach}(\mathcal{M}, \{s\}) \leq ub_{100} = 0.818181818181821948$$

holds where the difference $ub_{100} - lb_{100}$ is below 10^{-15} . The total computational effort for obtaining this precise result is about 92 seconds. If reduced accuracy suffices then runtime obviously improves. For instance, the fact

$$0.817971 = lb_{20} \leq \text{MaxReach}(\mathcal{M}, \{s\}) \leq ub_{20} = 0.818243$$

with $ub_{20} - lb_{20} < 10^{-3}$ was deduced within one second. With regard to the safety verification problem 4.1, system safety for each threshold value θ with $\theta < 0.817971$ or $\theta \geq 0.818243$ is *falsified* or *verified*, respectively, within a second.

With respect to competitive and more established methods based on value or policy iteration, we observed that the runtime of our prototypic tool chain does not compare favorably on the simple probabilistic reachability problem above. For instance, the version 4.0.1 of the PRISM model checker¹⁰ [KNP11] solved the problem in about 600 milliseconds with a precision of 10^{-15} (returning the result 0.8181818181818175).

In spite of the above fact, we have identified two promising directions for future research where probabilistic reachability analysis based on generalized Craig interpolation may pay off:

- (1) Embedding the same interpolation process into SSMT [FHT08], i.e. an arithmetic extension of SSAT, renders the generalized Craig interpolation scheme 4.3 *directly* applicable to probabilistic *hybrid* discrete-continuous systems, yielding a symbolic overapproximation of the backward reachable state set. As for the finite state case, scheme 4.4 then facilitates computing upper bounds of the reachability probability for hybrid systems by means of SSMT solving, just as already pursued when computing lower bounds according to the PBMC scheme 4.2 [FHT08, TF08, FTE10, TEF11].

It is important to remark that classical value or policy iteration procedures are *not* directly applicable in the hybrid state case but even after finite-state abstraction, confer, for instance, [ZSR⁺10, FHH⁺11].

- (2) Due to its *symbolic* nature, the analysis procedures based on SSAT and SSMT support *compact* representations of *concurrent* probabilistic (finite-state and hybrid) systems without an explicit construction of the product automaton [TEF11], the latter being of size exponential in the number of parallel components. This fact constitutes a strong argument that these symbolic procedures are able to alleviate the state explosion problem, which arises necessarily when applying explicit-state algorithms or methods based on finite-state abstraction refinement.

¹⁰More information can be found on <http://www.prismmodelchecker.org/>.

4.2. Interpolation-based probabilistic region stability. In addition to probabilistic state reachability being investigated in the previous section, we now address the problem of *probabilistic region stability*. For that purpose, we take into account the notion of *region stability* as introduced for non-probabilistic hybrid systems by Podelski and Wagner in [PW07a, PW07b]. According to their definition, given some set R of states called *region*, a (non-probabilistic) system is called *stable with respect to region R* iff for every infinite run $\langle s_0, s_1, \dots, s_i, \dots \rangle$ of the system, i.e. for every infinite sequence of states that follows the transition relation, there is some point of time $i \geq 0$ such that from i on the system remains in R forever, i.e. $\exists i \geq 0 \forall j \geq i : s_j \in R$.

Concerning the probabilistic case, several adaptations of region stability seem feasible, some of which pose measurability problems. Our main concern in this article being to identify potential application areas for generalized Craig interpolation rather than to discuss semantic issues of probabilistic stabilization, we do study a simple notion of probabilistic region stability in the sequel which circumvents measure-theoretic issues. As for probabilistic state reachability, we aim at defining a reasonable probability measure as the limit of the value of a recursive function defining the corresponding step-bounded measures. Intuitively, we consider finite run prefixes $\langle s_0, s_1, \dots, s_i \rangle$ such that from time point i on the probabilistic system remains in the given region forever under each possible future behavior, i.e. independent of the non-deterministic and probabilistic choices the system will take. The latter fact is guaranteed whenever the system has reached an invariance kernel of the given region that can never be left. The probability measure is then defined by the minimum probability of reaching the maximal invariance kernel.

Formally, let be given an MDP \mathcal{M} and a set of states $Region \subseteq S$ called the *stabilization region* or the *region* for short. An *invariance kernel* $\mathcal{K} \subseteq Region$ with respect to \mathcal{M} is a set of states from $Region$ such that there is no transition from a state in \mathcal{K} to a state outside \mathcal{K} , i.e. there does not exist a tuple $(z, act, z') \in \mathcal{K} \times Act \times (S \setminus \mathcal{K}) : ps(z, act, z') > 0$. An invariance kernel \mathcal{K} is called *maximal* if adding any new states to \mathcal{K} does not lead to an invariance kernel, i.e. each $\mathcal{K} \cup Z$ with $Z \subseteq Region \setminus \mathcal{K}$ and $Z \neq \emptyset$ is not an invariance kernel. Note that the maximal invariance kernel is unique. The latter fact can be simply shown using the observation that the set of all invariance kernels $\mathcal{K} \subseteq Region$ with respect to \mathcal{M} is closed under union. Let $\mathcal{K}^* \subseteq Region$ be the (unique) maximal invariance kernel with respect to \mathcal{M} . Then, the minimum probability $MinStable(\mathcal{M}, Region)$ that \mathcal{M} is *stable with respect to Region* is defined as the limit of the minimum step-bounded probability of reaching the maximal invariance kernel \mathcal{K}^* , i.e.

$$MinStable(\mathcal{M}, Region) = \lim_{k \rightarrow \infty} MinReach_{\mathcal{M}, \mathcal{K}^*}^k(i)$$

where

$$MinReach_{\mathcal{M}, \mathcal{K}^*}^k(s) = \begin{cases} 1 & ; s \in \mathcal{K}^* \\ 0 & ; s \notin \mathcal{K}^*, k = 0 \\ \min_{a \in Act} \sum_{s' \in S} ps(s, a, s') \cdot MinReach_{\mathcal{M}, \mathcal{K}^*}^{k-1}(s') & ; s \notin \mathcal{K}^*, k > 0 \end{cases}$$

gives the minimum probability of reaching \mathcal{K}^* from state $s \in S$ within k steps ($k \in \mathbb{N}$) under each possible scheduler.

When considering stabilization within *Region* as the *desired property* then the value of $MinStable(\mathcal{M}, Region)$ establishes the probability of stabilizing in *worst case*, i.e. under an optimal adversarial scheduler. For some threshold value $\theta \in [0, 1]$, the *stability verification problem* then is to decide whether this worst-case probability is at least θ , i.e. to decide

whether

$$\text{MinStable}(\mathcal{M}, \text{Region}) \geq \theta \quad (4.5)$$

holds.

In what follows, we propose a symbolic *verification* procedure for above problem 4.5. In a first phase, we compute a symbolic representation of an invariance kernel by means of generalized Craig interpolation. The main idea here is to iteratively eliminate states z not belonging to an invariance kernel from *Region* until a fixed point is reached. Due to the use of interpolation, the set of such states z is overapproximated in each iteration, meaning that potentially too many states are removed. This implies that the resulting invariance kernel is *not* necessarily maximal. However, each invariance kernel can be used for computing valid lower bounds of $\text{MinStable}(\mathcal{M}, \text{Region})$. The latter computation then is performed in a second phase by means of SSAT-based bounded reachability checking. Once a lower bound $lb \geq \theta$ is computed, property 4.5 is verified.

Phase 1. Let be given an SSAT encoding of an MDP \mathcal{M} as explained in Section 4.1 as well as some propositional formula $\text{Region}(\mathbf{s})$ encoding the stabilization region *Region*. Then, the state-set predicate $\mathcal{R}^k(\mathbf{s})$ for $k \in \mathbb{N}$ over state variables \mathbf{s} is inductively defined as

- $\mathcal{R}^0(\mathbf{s}) := \text{Region}(\mathbf{s})$, and
- $\mathcal{R}^{k+1}(\mathbf{s}) := \mathcal{R}^k(\mathbf{s}) \wedge \neg \mathcal{I}^{k+1}(\mathbf{s})$

where $\mathcal{I}^{k+1}(\mathbf{s}_{j-1})$ is a generalized Craig interpolant for

$$\left(\overbrace{\text{Trans}(\mathbf{s}_{j-1}, \mathbf{t}_j, \mathbf{s}_j) \wedge \neg \mathcal{R}^k(\mathbf{s}_j)}^{=A}, \overbrace{\text{Init}(\mathbf{s}_0) \wedge \bigwedge_{i=1}^{j-1} \text{Trans}(\mathbf{s}_{i-1}, \mathbf{t}_i, \mathbf{s}_i)}^{=B} \right)$$

with $j \geq 1$ with respect to SSAT formula

$$\mathcal{Q}(j) : \left(\overbrace{\text{Init}(\mathbf{s}_0) \wedge \bigwedge_{i=1}^{j-1} \text{Trans}(\mathbf{s}_{i-1}, \mathbf{t}_i, \mathbf{s}_i)}^{j-1 \text{ steps "forward" } (=B)} \wedge \overbrace{\text{Trans}(\mathbf{s}_{j-1}, \mathbf{t}_j, \mathbf{s}_j) \wedge \neg \mathcal{R}^k(\mathbf{s}_j)}^{\text{one step "backward" from } \neg \mathcal{R}^k (=A)} \right). \quad (4.6)$$

Observe that each $\mathcal{I}^{k+1}(\mathbf{s})$ can be computed by interpolating S-resolution if we rewrite $\neg \mathcal{R}^k(\mathbf{s})$ into CNF, the latter being always possible in linear time by adding auxiliary V_A -variables. During computation of each $\mathcal{I}^{k+1}(\mathbf{s})$, we take $I = \text{true}$ in every application of rule **RI.2** such that $\mathcal{I}^{k+1}(\mathbf{s})$ overapproximates all system states directly leading to the state set $\neg \mathcal{R}^k(\mathbf{s})$ due to Corollary 3.6. As a consequence, from each state in $\mathcal{R}^{k+1}(\mathbf{s}) = \mathcal{R}^k(\mathbf{s}) \wedge \neg \mathcal{I}^{k+1}(\mathbf{s})$ it is infeasible to leave the set $\mathcal{R}^k(\mathbf{s})$ in one step. Whenever the chain $\mathcal{R}^k(\mathbf{s})$ has stabilized, i.e.

$$\mathcal{R}^k(\mathbf{s}) \Rightarrow \mathcal{R}^{k+1}(\mathbf{s}),$$

it follows that $\mathcal{K}(\mathbf{s}) := \mathcal{R}^k(\mathbf{s})$ is an invariance kernel of $\text{Region}(\mathbf{s})$ with respect to \mathcal{M} , i.e. once entered, the system cannot leave the set $\mathcal{K}(\mathbf{s})$. Obviously, the chain $\mathcal{R}^k(\mathbf{s})$ eventually stabilizes in the finite-state case.

Similar to scheme 4.3, parameter $j \geq 1$ can be chosen arbitrarily, i.e. the system may execute any number of transitions until state \mathbf{s}_{j-1} is reached since this does not destroy the overapproximation property of $\mathcal{I}^{k+1}(\mathbf{s})$. The presence of parameter j gives us additional freedom in constructing generalized interpolants as j may influence the shape of $\mathcal{I}^{k+1}(\mathbf{s})$, as we will see in the example below.

Phase 2. Having computed a symbolic representation $\mathcal{K}(s)$ of a (not necessarily maximal) invariance kernel \mathcal{K} with respect to \mathcal{M} , we now compute lower bounds of the minimum probability $MinStable(\mathcal{M}, Region)$ of stabilizing within $Region$ by means of SSAT solving. To this end, first observe that $MinReach_{\mathcal{M}, \mathcal{K}^*}^k(\iota)$ is monotonic in k which implies that $MinReach_{\mathcal{M}, \mathcal{K}^*}^k(\iota) \leq MinStable(\mathcal{M}, Region)$ for each $k \in \mathbb{N}$. Let \mathcal{K}^* be the unique maximal invariance kernel with respect to \mathcal{M} . Then, $\mathcal{K} \subseteq \mathcal{K}^*$ since \mathcal{K} is an invariance kernel and the maximal invariance kernel \mathcal{K}^* is unique. As a consequence,

$$MinReach_{\mathcal{M}, \mathcal{K}}^k(\iota) \leq MinReach_{\mathcal{M}, \mathcal{K}^*}^k(\iota)$$

for each $k \in \mathbb{N}$. Summing up, each value of $MinReach_{\mathcal{M}, \mathcal{K}}^k(\iota)$ establishes a lower bound of $MinStable(\mathcal{M}, Region)$. In principle, $MinReach_{\mathcal{M}, \mathcal{K}}^k(\iota)$ can be reduced to an SSAT formula similar to PBMC scheme 4.2. The difference, however, is that we need to *minimize* the satisfaction probability. The latter can be achieved by a very similar SSAT encoding scheme that exploits *universal* quantifiers to resolve non-deterministic transition choices. Universal quantifiers then aim at minimizing the satisfaction probability. Though the SSMT solver SiSAT actually supports universal quantification, confer [TF09, TEF11], we instead stay within the scope of the logic exposed in this article and rephrase minimum probabilistic state reachability as a *maximum probabilistic state avoidance problem* as follows:

$$MaxAvoid_{\mathcal{M}, \mathcal{K}}^k(s) = \begin{cases} 0 & ; s \in \mathcal{K} \\ 1 & ; s \notin \mathcal{K}, k = 0 \\ \max_{a \in Act} \sum_{s' \in S} ps(s, a, s') \cdot MaxAvoid_{\mathcal{M}, \mathcal{K}}^{k-1}(s') & ; s \notin \mathcal{K}, k > 0 \end{cases}$$

It then holds that

$$MinReach_{\mathcal{M}, \mathcal{K}}^k(\iota) = 1 - MaxAvoid_{\mathcal{M}, \mathcal{K}}^k(\iota)$$

which can be proven by straightforward induction over step bound k . In the base cases, i.e. if $k = 0$ and $s \in \mathcal{K}$ or $s \notin \mathcal{K}$, the statement is clear. Within the induction step, we exploit the property that

$$\min_i \sum_j p_{i,j} \cdot P_{i,j} = 1 - \max_i \sum_j p_{i,j} \cdot (1 - P_{i,j})$$

is true for $0 \leq P_{i,j} \leq 1$ and $\sum_j p_{i,j} = 1$.

The problem of computing the value of $MaxAvoid_{\mathcal{M}, \mathcal{K}}^k(\iota)$ can be reduced to computing the maximum probability of satisfaction of the SSAT formula

$$\Phi_{\mathcal{M}, \mathcal{K}}^k = \mathcal{Q}(k) : \left(\overbrace{Init(s_0) \wedge \bigwedge_{i=1}^k Trans(s_{i-1}, t_i, s_i)}^{\text{states reachable within } k \text{ steps}} \wedge \overbrace{\bigwedge_{i=0}^k \neg \mathcal{K}(s_i)}^{\text{avoid invariance kernel}} \right).$$

According to the definition of $MaxAvoid_{\mathcal{M}, \mathcal{K}}^k(\iota)$, the propositional formula of $\Phi_{\mathcal{M}, \mathcal{K}}^k$ describes all system runs avoiding the invariance kernel \mathcal{K} for at least k transition steps. That is, all assignments encoding such latter runs yield satisfaction probability 1, while assignments encoding runs that visit \mathcal{K} within the first k steps do not satisfy the propositional formula, thus leading to satisfaction probability 0. As a consequence, $MaxAvoid_{\mathcal{M}, \mathcal{K}}^k(\iota) = Pr(\Phi_{\mathcal{M}, \mathcal{K}}^k)$.

j	\mathcal{I}^1	\mathcal{R}^1	\mathcal{I}^2	\mathcal{R}^2	\mathcal{K}
1	true	false	true	false	false
	$\{i, f, e, s\}$	\emptyset	$\{i, f, e, s\}$	\emptyset	\emptyset
2	true	false	true	false	false
	$\{i, f, e, s\}$	\emptyset	$\{i, f, e, s\}$	\emptyset	\emptyset
3	$\neg s$	$\neg f \wedge s$	$\neg s$	$\neg f \wedge s$	$\neg f \wedge s$
	$\{i, f, e\}$	$\{s\}$	$\{i, f, e\}$	$\{s\}$	$\{s\}$
4	$\neg s$	$\neg f \wedge s$	$\neg s$	$\neg f \wedge s$	$\neg f \wedge s$
	$\{i, f, e\}$	$\{s\}$	$\{i, f, e\}$	$\{s\}$	$\{s\}$

Table 2: Experimental results of applying the generalized interpolation scheme 4.6 on \mathcal{M} from Figure 4 for different values of parameter j . In addition to the symbolic representations computed by interpolation, the concrete state sets represented by these predicates are stated explicitly.

Using above facts, we deduce the following relation

$$\begin{aligned}
1 - Pr\left(\Phi_{\mathcal{M}, \mathcal{K}}^k\right) &= 1 - MaxAvoid_{\mathcal{M}, \mathcal{K}}^k(\iota) \\
&= MinReach_{\mathcal{M}, \mathcal{K}}^k(\iota) \\
&\leq MinReach_{\mathcal{M}, \mathcal{K}^*}^k(\iota) \\
&\leq MinStable(\mathcal{M}, Region) .
\end{aligned}$$

This finally enables us to compute lower bounds lb_k of $MinStable(\mathcal{M}, Region)$ using the scheme

$$lb_k := 1 - Pr\left(\Phi_{\mathcal{M}, \mathcal{K}}^k\right) , \quad (4.7)$$

the latter being addressed by SSAT solving. Note that the system behavior encoded by $\Phi_{\mathcal{M}, \mathcal{K}}^k$ becomes more and more constrained for increasing k such that the satisfaction probabilities $Pr\left(\Phi_{\mathcal{M}, \mathcal{K}}^k\right)$ are monotonically decreasing. This in turn means that the lb_k 's are monotonically increasing. With regard to solving the stability verification problem 4.5, the desired property $MinStable(\mathcal{M}, Region) \geq \theta$ is *verified* by the procedure above once a lower bound $lb_k \geq \theta$ is computed for some k .

Example. To illustrate the symbolic approach to probabilistic region stability based on generalized Craig interpolation, again consider the simple MDP \mathcal{M} from Figure 4 where the symbolic representation of the region is given by $Region(s) = \neg f$. That is, the region in which \mathcal{M} should stabilize consists of the states i , e , and s . The symbolic SSAT encoding of \mathcal{M} being introduced in the example of Section 4.1 is reused in the following.

We are first interested in computing an invariance kernel $\mathcal{K} \subseteq Region(s)$ with respect to \mathcal{M} by means of the generalized Craig interpolation scheme 4.6. To cope with the latter scheme automatically, we employ the simple interpolating DPLL-based SSAT solver mentioned in Section 4.1. The results of these experiments for different values of j are shown in Table 2. It is not hard to see that the unique maximal invariance kernel \mathcal{K}^* consists of the state s only. Recall that each interpolant \mathcal{I}^{k+1} overapproximates all system states directly leading to the state set $\neg \mathcal{R}^k$. When setting parameter j to value 1 or 2, we observe that interpolant $\mathcal{I}^1 = \text{true}$ is too coarse since it includes the whole state space. This causes the

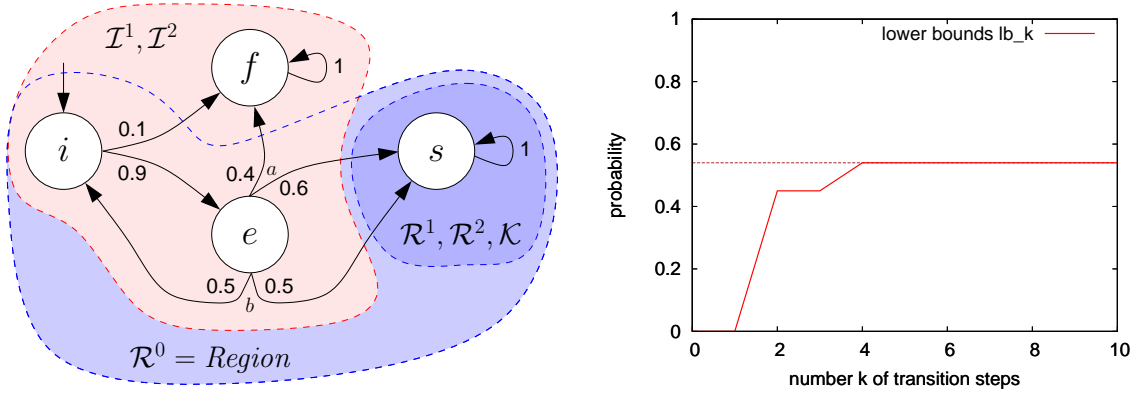


Figure 6: Illustration of the computed state sets for MDP \mathcal{M} by the generalized interpolation scheme 4.6 with $j \in \{3, 4\}$ (left), and lower bounds lb_k of the minimum probability of reaching the invariance kernel $\mathcal{K} = \{s\}$ over number k of transition steps computed by scheme 4.7 (right).

trivial invariance kernel $\mathcal{K} = \text{false}$ representing the empty set. For choices $j = 3$ and $j = 4$, however, $\mathcal{I}^1 = \neg s$ describes the exact set of states which lead to $\neg \mathcal{R}^0 = \neg \text{Region}$. Finally, the non-trivial invariance kernel $\mathcal{K} = \neg f \wedge s$ consisting of state s only is computed. Note that \mathcal{K} actually is the maximal invariance kernel. The computed state sets for $j \in \{3, 4\}$ are illustrated on the left of Figure 6.

These results confirm the observation made from the experiments of Section 4.1, namely that the greater the value of j , i.e. the more transition steps are performed, the more accurate the resulting overapproximations. Concerning runtime, each generalized Craig interpolant was computed by the interpolating DPLL-based SSAT solver within fractions of a second, where the highest runtime of 88 milliseconds was observed when computing \mathcal{I}^2 for $j = 4$.

Having computed an invariance kernel $\mathcal{K}(s) \subseteq \text{Region}(s)$ with respect to \mathcal{M} , we are now able to compute lower bounds lb_k of the minimum probability that \mathcal{M} is stable with respect to Region by means of scheme 4.7, where we use $\mathcal{K}(s) = \neg f \wedge s$ as obtained for $j \in \{3, 4\}$. Employing the SSMT tool SiSAT, some of the results are $lb_0 = lb_1 = 0$, $lb_2 = lb_3 = 0.45$, $lb_4 = lb_5 = 0.54$, \dots , $lb_{100} = 0.54$. Concerning runtime, all 100 SSAT formulae were solved within 88.16 seconds, while computation of the first 20 lower bounds lb_0 to lb_{20} just needed 600 milliseconds. The highest computation time for a single SSAT problem was obtained for lb_{100} , namely 2.91 seconds. The evolution of the lb_k 's up to $k = 10$ is presented graphically on the right of Figure 6. With regard to the stability verification problem 4.5, the desired property $\text{MinStable}(\mathcal{M}, \text{Region}) \geq \theta$ is *verified* for each threshold value $\theta \leq 0.54$ within a second.

Concerning competitive approaches, we remark that the probabilistic model checking tool PRISM 4.0.1 [KNP11] is also able to deal with probabilistic region stability of MDPs by means of path operators.¹¹ To determine the value of $\text{MinStable}(\mathcal{M}, \text{Region})$ for the example above, we used the specification $\text{Pmin=?} [\text{FP}>=1 [\text{G}(!f)]]$ meaning that we are interested in the minimum probability (Pmin=?) that finally (F) the system satisfies almost

¹¹Confer <http://www.prismmodelchecker.org/manual/PropertySpecification/ThePOOperator> for more detailed information.

surely ($P \geq 1$) the property that globally (G) state f is never visited ($!f$). PRISM solved the problem in 644 milliseconds returning the result 0.54.

As discussed for the case of probabilistic state reachability at the end of Section 4.1, we are also confident that the presented approach to probabilistic region stability based on generalized Craig interpolation becomes beneficial when adapted to probabilistic *hybrid* systems, where the classical procedures are not directly applicable. Furthermore, a particular pay-off is expected when dealing with *concurrent* probabilistic systems owing to the *symbolic* nature of the interpolation-based technique.

5. CONCLUSION AND FUTURE WORK

In this article, we elaborated on the idea of Craig interpolation for stochastic Boolean satisfiability. In consideration of the difficulties that arise in this stochastic extension of the propositional satisfiability problem, we first proposed a suitable definition of a generalized Craig interpolant and then presented an algorithm for automatically computing such interpolants. For the latter purpose, we enhanced the SSAT resolution calculus by corresponding rules for the construction of generalized Craig interpolants. We furthermore demonstrated two applications of generalized Craig interpolation as a means of automated analysis of probabilistic finite-state systems.

We first considered probabilistic state reachability. The resulting procedure is able to verify probabilistic safety requirements of the form “the worst-case probability of reaching undesirable system states is at most some given safety threshold”. This complements the existing SSAT-based probabilistic bounded model checking approach, which mechanizes falsification of such safety properties. As a second application, we gave attention to probabilistic region stability and presented a symbolic technique for verifying stability properties like “the worst-case probability that the system stabilizes within some given region is at least some given safety threshold”.

For future work, we are particularly interested in the adaptation of generalized Craig interpolation to SSMT, i.e. the extension of SSAT with arithmetic theories. One of the most challenging issues here will be the enhancement of the SSAT resolution calculus as well as the corresponding rules for the construction of generalized interpolants in order to deal with SSMT problems. The ability of computing generalized Craig interpolants for SSMT would lift the interpolation schemes 4.3 and 4.6 to SSMT problems, thus establishing symbolic verification approaches to probabilistic state reachability and to probabilistic region stability for discrete-time probabilistic hybrid systems. We are confident that such symbolic procedures will prove beneficial within the analysis of probabilistic hybrid systems, in particular when systems with a high degree of concurrency are considered.

ACKNOWLEDGEMENT

The authors wish to acknowledge fruitful discussions with the researchers in the AVACS project as well as in the MoVeS project, in particular with Andreas Eggers. Furthermore, we would like to thank the anonymous reviewers for their advice on how to enhance readability of the article.

REFERENCES

- [BCCZ99] Armin Biere, Alessandro Cimatti, Edmund M. Clarke, and Yunshan Zhu. Symbolic model checking without BDDs. In Rance Cleaveland, editor, *Proceedings of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems, TACAS '99*, volume 1579 of *Lecture Notes in Computer Science*, pages 193–207. Springer, 1999.
- [Bel57] Richard Bellman. A Markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957.
- [BHKH05] Christel Baier, Holger Hermanns, Joost-Pieter Katoen, and Boudewijn R. Haverkort. Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes. *Theor. Comput. Sci.*, 345(1):2–26, 2005.
- [BHvMW09] Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, February 2009.
- [BKF95] Hans Kleine Büning, Marek Karpinski, and Andreas Flögel. Resolution for quantified Boolean formulas. *Inf. Comput.*, 117(1):12–18, 1995.
- [BS06] Thanasis Balafoutis and Kostas Stergiou. Algorithms for stochastic CSPs. In Frédéric Benhamou, editor, *Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming (CP 2006)*, volume 4204 of *Lecture Notes in Computer Science*, pages 44–58. Springer, 2006.
- [BS07] Lucas Bordeaux and Horst Samulowitz. On the stochastic constraint satisfaction framework. In *Proceedings of the 2007 ACM Symposium on Applied Computing (SAC)*, pages 316–320. ACM, 2007.
- [BSST09] Clark Barrett, Roberto Sebastiani, Sanjit A. Seshia, and Cesare Tinelli. Satisfiability modulo theories. In Biere et al. [BHvMW09], chapter 26, pages 825–885.
- [Cra57] William Craig. Linear reasoning. a new form of the Herbrand-Gentzen theorem. *J. Symb. Log.*, 22(3):250–268, 1957.
- [DLL62] Martin Davis, George Logemann, and Donald W. Loveland. A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397, 1962.
- [DP60] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, 1960.
- [FHH⁺11] Martin Fränzle, Ernst Moritz Hahn, Holger Hermanns, Nicolás Wolovick, and Lijun Zhang. Measurability and safety verification for stochastic hybrid systems. In *Proceedings of the 14th International Conference on Hybrid Systems: Computation and Control (HSCC 2011)*, pages 43–52, New York, NY, USA, 2011. ACM.
- [FHT08] Martin Fränzle, Holger Hermanns, and Tino Teige. Stochastic satisfiability modulo theory: A novel technique for the analysis of probabilistic hybrid systems. In Magnus Egerstedt and Bud Mishra, editors, *Proceedings of the 11th International Conference on Hybrid Systems: Computation and Control (HSCC 2008)*, volume 4981 of *Lecture Notes in Computer Science*, pages 172–186. Springer, 2008.
- [FTE10] Martin Fränzle, Tino Teige, and Andreas Eggers. Engineering constraint solvers for automatic analysis of probabilistic hybrid automata. *Journal of Logic and Algebraic Programming*, 79(7):436–466, 2010.
- [KNP11] Marta Kwiatkowska, Gethin Norman, and David Parker. Prism 4.0: Verification of probabilistic real-time systems. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, *Proceedings of the 23rd International Conference on Computer Aided Verification (CAV 2011)*, volume 6806 of *Lecture Notes in Computer Science*, pages 585–591. Springer, 2011.
- [Lit99] Michael L. Littman. Initial experiments in stochastic satisfiability. In *Proceedings of the 16th National Conference on Artificial Intelligence*, pages 667–672, 1999.
- [LMP01] Michael L. Littman, Stephen M. Majercik, and Toniann Pitassi. Stochastic Boolean satisfiability. *Journal of Automated Reasoning*, 27(3):251–296, 2001.
- [Maj04] Stephen M. Majercik. Nonchronological backtracking in stochastic Boolean satisfiability. In *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004)*, pages 498–507. IEEE Computer Society, 2004.
- [Maj09] Stephen M. Majercik. Stochastic Boolean satisfiability. In Biere et al. [BHvMW09], chapter 27, pages 887–925.

- [McM03] Kenneth L. McMillan. Interpolation and SAT-based model checking. In Warren A. Hunt Jr. and Fabio Somenzi, editors, *Proceedings of the 15th International Conference on Computer Aided Verification (CAV 2003)*, volume 2725 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2003.
- [McM05] Kenneth L. McMillan. Applications of Craig interpolants in model checking. In Nicolas Halbwachs and Lenore D. Zuck, editors, *Proceedings of the 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2005)*, volume 3440 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2005.
- [ML98] Stephen M. Majercik and Michael L. Littman. MAXPLAN: A new approach to probabilistic planning. In *Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems*, pages 86–93. AAAI, 1998.
- [ML03] Stephen M. Majercik and Michael L. Littman. Contingent planning under uncertainty via stochastic satisfiability. *Artificial Intelligence Special Issue on Planning with Uncertainty and Incomplete Information*, 147(1-2):119–162, 2003.
- [Pap85] Christos H. Papadimitriou. Games against nature. *J. Comput. Syst. Sci.*, 31(2):288–301, 1985.
- [Pud97] Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *Journal of Symbolic Logic*, 62(3):981–998, September 1997.
- [PW07a] Andreas Podelski and Silke Wagner. Region stability proofs for hybrid systems. In Jean-François Raskin and P. S. Thiagarajan, editors, *Proceedings of the 5th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS 2007)*, volume 4763 of *Lecture Notes in Computer Science*, pages 320–335. Springer, 2007.
- [PW07b] Andreas Podelski and Silke Wagner. A sound and complete proof rule for region stability of hybrid systems. In Alberto Bemporad, Antonio Bicchi, and Giorgio C. Buttazzo, editors, *Proceedings of the 10th International Workshop on Hybrid Systems: Computation and Control (HSCC 2007)*, volume 4416 of *Lecture Notes in Computer Science*, pages 750–753. Springer, 2007.
- [Rob65] John Alan Robinson. A machine-oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, 1965.
- [TEF11] Tino Teige, Andreas Eggers, and Martin Fränzle. Constraint-based analysis of concurrent probabilistic hybrid systems: An application to networked automation systems. *Nonlinear Analysis: Hybrid Systems*, 5(2):343–366, 2011.
- [TF08] Tino Teige and Martin Fränzle. Stochastic satisfiability modulo theories for non-linear arithmetic. In Laurent Perron and Michael A. Trick, editors, *Proceedings of the 5th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR 2008)*, volume 5015 of *Lecture Notes in Computer Science*, pages 248–262. Springer, 2008.
- [TF09] Tino Teige and Martin Fränzle. Constraint-based analysis of probabilistic hybrid systems. In Alessandro Giua, Cristian Mahulea, Manuel Silva, and Janan Zaytoon, editors, *Proceedings of the 3rd IFAC Conference on Analysis and Design of Hybrid Systems*, pages 162–167. IFAC, 2009.
- [TF10] Tino Teige and Martin Fränzle. Resolution for stochastic Boolean satisfiability. In Christian G. Fermüller and Andrei Voronkov, editors, *Proceedings of the 17th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR-17)*, volume 6397 of *Lecture Notes in Computer Science*, pages 625–639. Springer, 2010.
- [Wal02] Toby Walsh. Stochastic constraint programming. In Frank van Harmelen, editor, *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002)*, pages 111–115. IOS Press, 2002.
- [ZSR⁺10] Lijun Zhang, Zhikun She, Stefan Ratschan, Holger Hermanns, and Ernst Moritz Hahn. Safety verification for probabilistic hybrid systems. In Tayssir Touili, Byron Cook, and Paul Jackson, editors, *Proceedings of the 22nd International Conference on Computer Aided Verification, CAV 2010*, volume 6174 of *Lecture Notes in Computer Science*, pages 196–211. Springer, 2010.